کمپیوٹر سائنس

نہم دہم

حصہ دوئم

تیار کردہ: پنجاب ٹیکسٹ بک بورڈ، لاہور۔
منظور کردہ: وفاقی وزارت تعلیم، کریکولم ونگ، اسلام آباد۔
بحوالہ چٹھی نمبر E1-10/2005-Maths (Comp. Sc.). Dated July 8, 2006

فہرست

مصنفین:

☆ سید ذوالفقر نین جعفری
اسسٹنٹ پروفیسر
COMSATS انسٹیٹیوٹ آف انفارمیشن ٹیکنالوجی
سیکٹر H-8، اسلام آباد

☆ مسز شاہینہ
صدر شعبہ (شعبہ کمپیوٹر سائنس)
اسلام آباد ماڈل کالج فار گرلز
F-10/2، اسلام آباد

☆ آصف علی عباسی
لیکچرار (کمپیوٹر سائنس)
اسلام آباد کالج فار بوائز، G-3
اسلام آباد

مترجمین:

☆ ڈاکٹر محمد نعیم گل
ایسوسی ایٹ پروفیسر
یونیورسٹی آف انجینئرنگ اینڈ ٹیکنالوجی، لاہور

☆ مقصود رضا
ایسوسی ایٹ پروفیسر
گورنمنٹ شالیمار کالج، باغبانپورہ، لاہور

ماہر زبان:

☆ پروفیسر محمود علی انجم
ایم۔اے/ایم۔سی۔ایس
دارالمصنفین، طارق آباد، فیصل آباد

ایڈیٹر:

☆ مرزا مبشر بیگ
ریسرچ ایسوسی ایٹ (ڈیپارٹمنٹ آف کمپیوٹر سائنس)
لاہور یونیورسٹی آف مینجمنٹ سائنسز (LUMS)، لاہور

نگران طباعت

☆ مظہر حیات
ماہر مضمون پنجاب ٹیکسٹ بک بورڈ، لاہور

باب 1

مسائل حل کرنا

(Problem Solving)

1.1 تعارف (Introduction)

ہم مسائل حل کرتے ہیں اور ہر روز گھر پر، کام کے دوران، کھیل کے دوران اور حتیٰ کہ مختلف اشیاء کی خریداری کے وقت فیصلے کرتے ہیں۔ کچھ مسائل اور فیصلے بہت کٹھن اور توجہ طلب ہوتے ہیں اور اُن کے لیے بہت سوچ و بچار، جذبے اور تحقیق کی ضرورت ہوتی ہے۔ تاہم، مسئلہ کی نوعیت کچھ بھی ہو، ہم ہمیشہ کئی حل تلاش کرنے کی کوشش کرتے ہیں تاکہ ہم ان میں سے بہترین حل منتخب کر سکیں۔

1.2 مسائل حل کرنے کا طریقہ (Problem Solving Method)

مسائل حل کرنا ایک مہارت ہے جو کہ ایک بہت منظّم طریقۂ کار اختیار کرنے سے پیدا کی جاسکتی ہے۔ پروگرامنگ (Programming) بھی مسئلہ حل کرنے کی ہی ایک سرگرمی ہے۔ اگر آپ مسائل حل کرنے میں مہارت رکھتے ہیں تو آپ میں ایک اچھا پروگرامر (Programmer) بننے کی صلاحیت موجود ہے۔ مسائل حل کرنے کے طریقے بہت سے مضامین میں سکھلائے جاتے ہیں۔

بزنس (Business) کے طالبعلم متعلقہ سسٹم اَپروچ (System Aproach) سے مسائل حل کرنا سیکھتے ہیں جبکہ انجینئرنگ اور سائنس کے طالبعلم انجینئرنگ اور سائنسی طریقوں کو اِستعمال کرتے ہیں۔ پروگرامرز سوفٹ ویئر (Programmer's Software) بنانے کے طریقے اِستعمال کرتے ہیں۔ کسی بھی مسئلہ کو حل کرنے کے لیے مندرجہ ذیل اِقدام اٹھائے جاسکتے ہیں:

1۔ مسئلہ کا تعین
2۔ ضرورتیں وضاحت سے بیان کرنا
3۔ مسئلہ کا تجزیہ کرنا
4۔ الگورتھم (Algorithm) اور فلوچارٹ (Flowchart) بنانا
5۔ پروگرام لکھنا (کوڈنگ - Coding)
6۔ پروگرام کو ٹیسٹ کرنا اور اس کی ایررز (Errors) درست کرنا (ڈی بگنگ - Debugging)
7۔ پروگرام استعمال کرنا (لاگو کرنا)
8۔ پروگرام کی دیکھ بھال کرنا اور بہتر بنانا
9۔ پروگرام کو ڈاکیومنٹ (Document) کرنا

1.2.1 مسئلہ کا تعین (Problem Identification)

اس مرحلہ پر زیرِ حل مسئلہ کا بغور مشاہدہ کیا جاتا ہے، متعلقہ امور کا تعین کیا جاتا ہے اور غیر متعلقہ معلومات حذف کر دی جاتی ہیں۔ فرض کریں ہم سادہ کیلکولیٹر (Calculator) بنانا چاہتے ہیں۔ ہمارا اصل مسئلہ یہ ہے کہ بنیادی حسابی مراحل (جمع، تفریق، ضرب اور تقسیم) کیسے سرانجام دیے جاتے ہیں؟ نتیجہ کس طرح ظاہر ہونا چاہیے؟ ان پٹ کس طرح لینا چاہیے وغیرہ وغیرہ؟ ہمیں اس بات میں

دلچسپی نہیں ہے کہ سائن (Sine) اور ٹین (Tan) کیسے معلوم کئے جاتے ہیں؟ الجبری مساوات کیسے حل کی جاتی ہے؟ چونکہ یہ باتیں ہمارے لیے غیر متعلقہ ہیں، اس لیے ہمیں ان کے بارے میں پریشان نہیں ہونا چاہیے۔ اس طریقے سے، غیر متعلقہ معلومات چھوڑ کر ہم اصل مسئلہ پر توجہ مرکوز کر سکتے ہیں۔

1.2.2 ضرورتیں وضاحت سے بیان کرنا (Specify Requirements)

بہت سے یوزرز (Users) اپنے سوفٹ ویئر کی صحیح ضروریات کی وضاحت نہیں کر سکتے۔ انہیں یقینی طور پر علم نہیں ہوتا کہ وہ سوفٹ ویئر سے کیا کام لینا چاہتے ہیں۔ لہٰذا ان کے ذہن میں ضرورتوں کا غیر واضح سیٹ ہوتا ہے جو کہ اُنہیں غلط حل کی طرف لے جا سکتا ہے۔ یہ مرحلہ یوزر کی ضروریات کو واضح کرنے کا تقاضا کرتا ہے تاکہ مناسب حل تجویز کیا جا سکے۔ اس مرحلہ میں ''ضرورتوں پر مشتمل دستاویز'' تیار کی جاتی ہے جو سسٹم کی متوقع خصوصیات بیان کرتی ہے، ان پابندیوں کا ذکر کرتی ہے جن میں اسے کام کرنا چاہیے، اس سوفٹ ویئر کا خیالی ذکر کرتی ہے جسے ڈیزائن کیا جائے گا اور استعمال میں لایا جائے گا۔

1.2.3 مسئلہ کا تجزیہ کرنا (Analyze the Problem)

اس مرحلہ میں مسئلہ کو چھوٹے مسائل میں تقسیم کیا جاتا ہے۔ مجموعی طور پر بڑے مسئلہ پر توجہ مرکوز کرنے کی بجائے ہم ہر تحتی مسئلہ کو الگ سے حل کرنے کی کوشش کرتے ہیں۔ اس سے سادہ حل نکل آتا ہے۔ یہ حکمتِ عملی ٹاپ ڈاؤن ڈیزائن (تقسیم کرو اور فتح کرو کا اصول بھی) کہلاتی ہے۔ صحیح حل تک پہنچنے کے لیے ہم چند سوالات پوچھ سکتے ہیں۔ مثال کے طور پر

1۔ دیئے گئے مسئلہ کے کتنے حل ہیں؟
2۔ کون سا حل بہترین ہے؟
3۔ کیا مسئلہ کو کمپیوٹر پر حل کیا جا سکتا ہے؟
4۔ ان پٹ اور آؤٹ پٹ کیا ہیں؟
5۔ زیادہ بڑے مسئلہ کو چھوٹے مسائل میں کیسے تقسیم کیا جا سکتا ہے؟

1.2.4 الگورتھم ڈیزائن کرنا اور فلوچارٹ بنانا (Design the Algorithm and Draw Flowchart)

الگورتھم ڈیزائن کرنا

مسئلہ کو حل کرنے کے لیے الگورتھم کی ڈیزائننگ کے لیے آپ کو مراحل کی فہرست دینا ہوتی ہے۔ پھر تصدیق کی جاتی ہے کہ الگورتھم مسئلہ کو حسب منشا حل کرتا ہے یا نہیں۔ الگورتھم لکھنا اکثر مسئلہ حل کرنے کے عمل کا مشکل ترین حصّہ ہوتا ہے۔ زیادہ تر کمپیوٹر الگورتھمز کم از کم مندرجہ ذیل تین اقدام سرانجام دیتے ہیں۔

1۔ ڈیٹا حاصل کرنا (ان پٹ - Input) 2۔ کمپیوٹیشن (Computation) کرنا (پروسیسنگ)
3۔ نتائج ظاہر کرنا (آوٹ پٹ - Output)

ایک بار الگورتھم بن جائے تو اس کی ڈیسک چیکنگ (Desk checking) کے ذریعے تصدیق کرنی چاہیے۔ ڈیسک چیکنگ الگورتھم ڈیزائن کا ایک اہم حصّہ ہے جس کو اکثر نظر انداز کر دیا جاتا ہے۔ ایک الگورتھم کو ڈیسک چیک کرنے کے لیے، ہمیں الگورتھم کے ہر

ایک مرحلہ پر اس طرح عمل کرنا چاہیے جس طرح کہ ایک کمپیوٹر کرتا ہے، اور تصدیق کرنی چاہیے کہ الگورتھم حسب منشا کام کرتا ہے۔ اس مرحلہ پر الگورتھم کو چیک کر کے اور اس کی ایررز چیک کر کے وقت اور کوشش بچائے جا سکتے ہیں۔

ایک پروگرام کسی خاص مسئلہ کو حل کرنے کے لیے کمپیوٹر کو دی گئی ہدایات کا مجموعہ ہوتا ہے۔

ڈیسک چیکنگ کسی ٹیسٹ ڈیٹا کی مدد سے کاغذ پر الگورتھم کے کام کے محتاط مشاہدہ کا عمل ہے۔ الگورتھم کو مختلف قیمتوں کی شکل میں ان پٹ دیا جاتا ہے جس کے آؤٹ پٹ کا جائزہ لیا جاتا ہے۔

فلوچارٹ بنانا

الگورتھم ڈیزائن کرنے کے بعد اگلا مرحلہ فلو چارٹ بنانے کا ہوتا ہے۔ فلو چارٹ درحقیقت تصویری شکل میں الگورتھم کی نمائندگی کرتا ہے جس سے الگورتھم میں کنٹرول کی سمت اور ڈیٹا کو سمجھنے میں مدد ملتی ہے۔ ہم اس باب میں بعد میں تفصیل سے فلوچارٹ بنانے کا طریقہ بیان کریں گے۔

1.2.5 پروگرام لکھنا (Write the Program or Coding)

یہ مرحلہ کسی بھی پروگرامنگ لینگوئج (Language) میں لکھے گئے الگورتھم کو پروگرام میں تبدیل کرنے پر مشتمل ہوتا ہے۔ اس مقصد کے لیے پروگرامر کو منتخب پروگرامنگ لینگوئج کے سینٹیکس (Syntax) کا علم ہونا چاہیے۔

کسی پروگرام لینگوئج میں پروگرام لکھنے کے اصول اس پروگرامنگ لینگوئج کا سینٹیکس کہلاتے ہیں۔

1.2.6 پروگرام کو ٹیسٹ کرنا اور اس کی ایررز (Errors) معلوم اور درست کرنا

(Test and Debug the Program)

اس مرحلے میں پروگرام کے حسب منشا کام کرنے کی تصدیق کے لیے اسے پرکھا جاتا ہے۔ صرف ایک ٹیسٹ کیس (Test case) پر اکتفا نہ کریں۔ مختلف ڈیٹا سیٹوں کو استعمال کرتے ہوئے پروگرام کو کئی بار چلائیں اور یہ یقین کر لیں کہ یہ الگورتھم کے لیے مہیا کی گئی ہر صورتحال میں صحیح کام کرتا ہے۔ اگر یہ مطلوبہ نتائج نہ دے رہا ہو تو ایررز (بگز) کی نشاندہی کرنی چاہیے اور انہیں درست کرنا چاہیے۔ ڈی بگنگ ایک ایسا عمل ہے جو پروگرام میں ایررز تلاش کرتا اور اُن کو دُور کرتا ہے۔ پروگرامنگ کی ایررز کی تین اقسام ہو سکتی ہیں: سینٹیکس ایررز، رن ٹائم (Time) ایررز اور لوجیکل (Logical) ایررز۔

پروگرام میں ایررز تلاش کرنے اور دور کرنے کے عمل کو ڈی بگنگ کہتے ہیں۔

سینٹیکس ایررز (Syntax Errors)

جب پروگرام، پروگرام لینگوئج کے ایک یا زائد گرائمر کے اصولوں کی خلاف ورزی کرتا ہے تو سینٹیکس کی ایرر واقع ہو جاتی ہے۔ ان ایررز کا کمپائلیشن (Compilation) کے وقت یعنی جب ٹرانسلیٹر (Translator) (کمپائلر - Compilar یا انٹرپریٹر - Interpreter) پروگرام کا ترجمہ کرنے کی کوشش کرتا ہے تو پتہ چلتا ہے۔ اس ایرر کی بہت سی وجوہات ہو سکتی ہیں، مثال کے طور پر ایک غلط پروگرام سٹیٹمنٹ (Statement) یا کمانڈ (Command) کو ایگزیکیوٹ (Execute) کرنے کے لیے ٹائپ

(Type) کرنا جیسا کہ PRINT سٹیٹمنٹ کی بجائے PINT ٹائپ کرنا یا مستقل مقدار (کانسٹنٹ ویلیو - Constant value) کو قیمت دینے کی کوشش کرنا جیسا کہ 5=count وغیرہ۔

کیا آپ جانتے ہیں کہ صفر پر تقسیم غیر تعریف شدہ ہے۔

رن ٹائم ایررز (Run Time Errors)

جب پروگرام کمپیوٹر کو کوئی غیر قانونی یا غیر تعریف شدہ کام کرنے کی ہدایت دیتا ہے تو رن ٹائم ایر رواقع ہو جاتی ہے، جیسا کہ کسی عدد کو صفر سے تقسیم کرنا۔ کمپیوٹر رن ٹائم ایررز کو پروگرام کی ایگزیکیوشن کے دوران معلوم اور ظاہر کرتا ہے۔ جب رَن ٹائم ایرر واقع ہوگی تو کمپیوٹر آپ کے پروگرام پر عمل درآمد کرنا بند کر دے گا اور ایرر کی نشاندہی پر مشتمل پیغام ظاہر کرے گا جو کہ ایرر کو ڈھونڈنے میں مدد دیتا ہے۔

منطقی ایررز (Logical Errors)

جب پروگرام ایک غلط الگورتھم کی پیروی کرتا ہے تو منطقی ایرر واقع ہو جاتی ہے۔ ٹرانسلیٹر منطقی ایرر کے لیے کوئی پیغام نہیں دیتا۔ ان ایررز کو ڈھونڈنا انتہائی مشکل ہے۔ آپ اپنے پروگرام کا غلط آؤٹ پُٹ دیکھ کر ہی منطقی ایرر کا اندازہ لگا سکتے ہیں۔ منطقی ایررز پروگرام کو مکمل طور پر ٹیسٹ کرتے ہوئے، تمام متغیرات پر قریب سے غور کرتے ہوئے اور پروگرام میں منطقی فلو کے ہر راستے کو ٹیسٹ کرتے ہوئے ڈھونڈی جا سکتی ہیں۔

1.2.7 پروگرام لاگو کرنا (Implement the Program)

ایک مرتبہ پروگرام مکمل طور پر ٹیسٹ ہو جانے کے بعد ایسی جگہ انسٹال (Instal) کرنا یا رکھنا چاہیے جہاں اسے استعمال کیا جائے گا۔ اس مرحلہ کو پروگرام کا عملی استعمال کہتے ہیں۔

1.2.8 پروگرام کی دیکھ بھال کرنا اور بہتر بنانا (Maintain and Update the Program)

پروگرام کی دیکھ بھال پروگرام کو بہتر بنانے کے لیے ایک ایسا جاری رہنے والا پروسیس (Process) ہے جو نئے ہارڈ ویئر (Hardware) یا سوفٹ ویئر کے تقاضوں کو پورا کرتا ہے۔ پروگرام کی انسٹالیشن (Installation) کے بعد یہ محض پروگرام کی وسعت، بہتری اور اصلاح کا عمل ہے۔ پروگرام کی ہمہ وقت افادیت کی خاطر مسلسل دیکھ بھال ضروری ہے اور مناسب دیکھ بھال کا دارومدار پروگرام کی تحریری شکل پر ہے۔

1.2.9 پروگرام ڈاکیومنٹ کرنا (Document the Program)

ڈاکیومنٹیشن (Documentation) پروگرام کے الگورتھم، ڈیزائن، کوڈنگ کے طریقہ، ٹیسٹنگ اور مناسب استعمال کی ایک تفصیلی وضاحت ہے۔ یوزر کے لیے جو کہ ہر روز پروگرام پر انحصار کرتا ہے یا اس پروگرامر کے لیے جس کو پروگرام میں تبدیلی کے لیے یا اسے بہتر بنانے کے لیے کہا جاتا ہے، ڈاکیومنٹیشن ضروری ہے۔

پروگرامز کی ڈاکیومنٹیشن میں کیا شامل کیا جائے سے متعلقہ عالمی سطح پر قابل قبول سٹینڈرڈز (Standards) نہیں ہیں۔ اگرچہ پروگرامز کی پیچیدگی کے لحاظ سے اس میں شامل چیزیں مشتمل ہو سکتی ہیں، عام طور پر، جامع ڈاکیومنٹیشن درج ذیل ریکارڈز پر مشتمل ہوتی ہے۔

☆ پروگرام نے کیا کرنا ہے سے متعلق وضاحت (سوفٹ ویئر کی ضرورت سے متعلقہ ڈاکیومنٹ)۔

☆ مسئلہ کے حل کی وضاحت (الگورتھم)۔

☆ پروگرام کے ڈیزائن کی وضاحت جس میں استعمال شدہ معاون اشیاء (فلوچارٹس، الگورتھم وغیرہ) شامل ہیں۔

☆ پروگرام کے ٹیسٹنگ (Testing) پروسیس کی وضاحت جس میں استعمال شدہ ٹیسٹ ڈیٹا اور حاصل کردہ نتائج شامل ہیں۔

☆ اس کے استعمال کے وقت سے کی گئی تمام درستگیوں، تبدیلیوں اور پروگرام میں کیے گئے اضافوں کی وضاحت شامل ہے۔

☆ یوزر مینوئل (User Manual) / یوزر گائیڈ (User guide)۔

1.3 الگورتھم (Algorithm)

الگورتھم مراحل کا ایک متناہی سیٹ ہے جس کی اگر پیروی کی جائے تو ایک خاص کام تکمیل تک پہنچتا ہے۔ الگورتھم واضح، حتمی اور موثر ہونا چاہیے۔

الگورتھم کی سادہ ترین شکل مرحلہ وار الگورتھم (مثلاً To-do لسٹ) ہے۔ یہ سلسلہ وار مراحل کی ترتیب پر مشتمل ہوتا ہے۔

1.3.1 الگورتھم ڈویلپ کرنے کی حکمت عملی (Strategy for Developing Alogrithm)

الگورتھم کی ڈویلپمنٹ (Development) کے لیے درج ذیل مراحل کو طے کرنا ہوتا ہے جس سے ہم کسی خاص مسئلہ کے درست حل کی طرف جا سکتے ہیں۔

مرحلہ 1: انویسٹی گیشن (Investigation)

(i) طریقہ کار کی شناخت کرنا۔ (ii) بڑے فیصلوں کی شناخت کرنا۔

(iii) تکرار کی شناخت کرنا۔ (iv) متغیرات کی شناخت کرنا۔

مرحلہ 2: ابتدائی الگورتھم

(i) ایک ہائی لیول (High level) / (جنرل) الگورتھم بنائیے۔

(ii) الگورتھم کے مطابق چلیے۔ کیا ایسا کرنے سے کسی بڑے مسئلہ کا علم ہوتا ہے؟ اگر ایسا ہے تو مسئلہ کو درست کیجیے۔

مرحلہ 3: الگورتھم کے مرحلہ کو ریفائن کرنا

(i) مرحلہ 2 میں نشاندہی کی گئی کسی بھی ریفائنمنٹ (Refinement) کو شامل کرنا۔

(ii) جہاں مناسب ہو عوامل کو گروپ کی شکل میں اکٹھا کرنا۔

(iii) جہاں مناسب ہو متغیرات کو گروپ کی شکل میں اکٹھا کرنا۔

(iv) الگورتھم کو مرحلوں کے ذریعے دوبارہ ٹیسٹ کرنا۔

آیئے ہم درج ذیل مثال سے کسی مسئلہ کے حل تک پہنچنے کے طریقہ کار کو سمجھتے ہیں۔

مسئلہ 1: آپ نے اپنے گھر پر کیک بیک (Bake) کرنا ہے۔

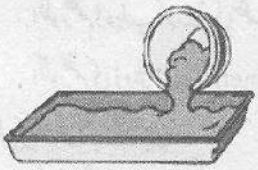
مرحلہ: الگورتھم بنائیے۔

درج ذیل اشکال کیک بنانے کے مراحل کو ظاہر کرتی ہیں۔

مرحلہ 1: اوون کو $325^{o}F$ تک گرم کیجیے۔

مرحلہ 2: تمام اجزا کو اکٹھا کیجیے۔

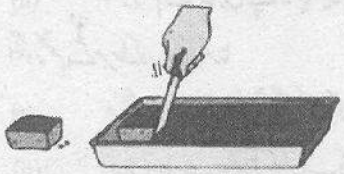
مرحلہ 3: اجزا کو ایک پیالہ میں اچھی طرح ملائیے۔



مرحلہ 4: ملے ہوئے اجزا کو بیکنگ کے برتن میں ڈالیے۔

مرحلہ 5: 50 منٹ تک اوون میں بیک کیجیے۔

مرحلہ 6: 5 منٹ مزید بیک کیجیے جب تک کہ کیک درمیان سے دبانے سے واپس نہ آئے۔



مرحلہ 7: کاٹنے سے پہلے اسے ایک ریک پر ٹھنڈا کیجیے۔

مسئلہ 2: آپ کو گھر پر چائے بنانا ہے۔

مرحلہ: الگورتھم بنائیے۔

الگورتھم پر ابتدائی کوشش یہ ہو سکتی ہے۔

1۔ چائے کی پتی برتن میں ڈالیے۔ 2۔ پانی اُبالیے۔

3۔ 5 منٹ تک انتظار کیجیے۔ 4۔ کپ میں چائے اُنڈیلیے۔

⇦ **پہلی ریفائنمنٹ:**

مندرجہ بالا مراحل غالباً چائے بنانے کے لیے کافی تفصیلی نہیں ہیں۔اس لیے ہم چند مراحل کو چھوٹے مرحلوں میں ترتیب وار ریفائن (Refine) کرتے ہیں۔

مرحلہ 1: برتن میں چائے کی پتی ڈالیے

یہ اس طرح ریفائن ہوسکتا ہے کہ:

1.1 چائے کی پتی کا ڈبہ کھولیے۔ 1.2 پتی سے بھرا ہوا ایک چائے کا چمچ نکالیے۔

1.3 پتی سے بھرا ہوا چمچ برتن میں ڈالیے۔ 1.4 چائے کا ڈبہ بند کیجیے۔

مرحلہ 2: پانی اُبالیے

یہ اس طرح ریفائن ہوسکتا ہے کہ

2.1 کیتلی کو پانی سے بھریئے۔ 2.2 کیتلی کا سوئچ (Switch) آن (On) کیجیے۔

2.3 پانی کے اُبلنے تک انتظار کیجیے۔ 2.4 کیتلی کا سوئچ آف (Off) کیجیے۔

مرحلہ 5: چائے کپ میں اُنڈیلیے۔

یہ اس طرح ریفائن ہوسکتا ہے۔

5.1 چائے کو برتن سے کپ میں اُنڈیلیے حتیٰ کہ کپ بھر جائے۔

⇦ **دوسری ریفائنمنٹ:**

ریفائن کیے گئے کچھ مراحل کو مزید ریفائن کیا جاسکتا ہے۔مثال کے طور پر

مرحلہ 2.1: کیتلی کو پانی سے بھریئے۔

یہ اس طرح ریفائن ہوسکتا ہے کہ:

2.1.1 کیتلی کو ٹونٹی کے نیچے رکھیے۔ 2.1.2 ٹونٹی کھولیے۔

2.1.3 کیتلی کے بھرنے تک انتظار کیجیے۔ 2.1.4 ٹونٹی بند کردیجیے۔

دوسرے مراحل کی مزید ریفائنمنٹ بھی درکار ہوسکتی ہے۔کافی ریفائنمنٹس کے بعد روبوٹ ہر مرحلہ کو ایگزیکیوٹ کرنے کے قابل ہوجاتا ہے۔

اورجنل الگورتھم	پہلی ریفائنمنٹ	دوسری ریفائنمنٹ
1۔ چائے کی پتی برتن میں ڈالیے۔	1.1 چائے کا ڈبہ کھولیے۔	1.1.1 شیلف (Shelf) سے چائے کا ڈبہ لیجیے۔
		1.1.2 ڈبہ کے ڈھکن کو ہٹایئے۔
	1.2 ایک چائے کا بھرا ہوا چمچ نکالیے۔	
	1.3 چائے سے بھرا ہوا چمچ برتن میں ڈالیے۔	

	1.4 چائے کا ڈبہ بند کیجیے۔	1.4.1 ڈبہ پر ڈھکن لگائیے۔ 1.4.2 ڈبہ کو دوبارہ شیلف پر رکھیے۔
2۔ پانی کو اُبالیے۔	2.1 کیتلی کو پانی سے بھریئے۔	2.1.1 کیتلی کو ٹونٹی کے نیچے رکھیے۔
		2.1.2 ٹونٹی کھولیے۔
		2.1.3 کیتلی کے بھرنے تک انتظار کیجیے۔
		2.1.4 ٹونٹی بند کر دیجیے۔
	2.2 کیتلی کا سوئچ آن کیجیے۔	
	2.3 پانی کے اُبلنے تک انتظار کیجیے۔	2.3.1 کیتلی کے سیٹی بجانے تک انتظار کیجیے۔
	2.4 کیتلی کا سوئچ آف کیجیے۔	
3۔ برتن میں اور پانی ڈالیے۔	3.1 کیتلی سے پانی اُنڈیلیے حتیٰ کہ برتن بھر جائے۔	
4۔ 5 منٹ انتظار کیجیے۔		
5۔ چائے کپ میں اُنڈیلیے۔	5.1 برتن سے کپ میں چائے اُنڈیلیے حتیٰ کہ کپ بھر جائے۔	

مثال1: پہلے پچاس قدرتی اعداد کا مجموعہ معلوم کرنے کے لیے الگورتھم لکھیں۔

```
Algorithm
BEGIN
    SUM = 0
    N = 0
    DO WHILE (N <= 50)
        SUM = SUM + N
        N = N + 1
    END DO
END
```

مثال2: دیے گئے عدد کا فیکٹوریل (Factorial) معلوم کرنے کے لیے الگورتھم لکھیں۔

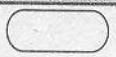
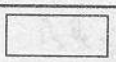
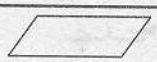
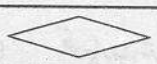
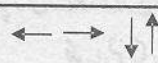
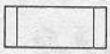
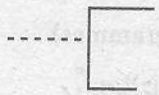
```
Algorithm
BEGIN
    fact = 1
    n = 1
    PRINT "Enter a number"
    INPUT num
    FOR n = 1 to num
        fact = fact * n
    NEXT n
    PRINT fact
END
```

1.4 فلوچارٹ (Flowchart)

فلوچارٹ الگورتھم کا بذریعہ تصاویر اظہار ہے۔ یہ ڈیٹا کو سسٹم میں ہونے والے عوامل اور ان پر عملدرامد کی ترتیب کو ظاہری شکل میں پیش کرنے کا ایک طریقہ ہے۔ فلوچارٹ کسی عمارت کے نقشے جیسا ہوتا ہے۔ جیسا کہ ہم جانتے ہیں کہ ایک ڈیزائنر (Designer) بلڈنگ تعمیر کرنے سے پہلے اُس کا نقشہ بناتا ہے۔ اسی طرح ایک پروگرامر کمپیوٹر پروگرام لکھنے سے پہلے فلوچارٹ بنانے کو ترجیح دیتا ہے۔ نقشہ بنانے کی طرح فلوچارٹ بھی واضح کردہ اصولوں کے مطابق بنایا جاتا ہے۔

1.4.1 فلوچارٹ کی علامات (Symbols of Flowchart)

فلوچارٹس کو عام طور پر خاص علامتیں (سمبلز - Symbols) استعمال کرتے ہوئے بنایا جاتا ہے، تاہم کچھ دیگر خاص علامتیں بھی ضرورت کے مطابق واضح کی جاسکتی ہیں۔ فلوچارٹنگ کے لیے اکثر استعمال ہونے والی علامات درج ذیل ہیں۔

مقصد	علامت
پروگرام کا آغاز یا اختتام	
پروسیسنگ (Processing)	
اِن پُٹ یا آؤٹ پُٹ	
فیصلہ کرنا اور پروگرام کے دو حصّوں کو ملانا	
کونیکٹر (Connector)	
آف پیج / آن پیج کونیکٹر	
فلولائنز (Flowlines)	
پری ڈیفائنڈ (Predefined) پروسیس (فنکشنز/سب روٹینز)	
ریمارکس (Remarks)	

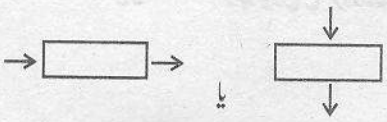
1.4.2 فلوچارٹ بنانے کے لیے گائیڈ لائنز (Guidelines for Drawing a Flowchart)

(i) ایک مناسب فلوچارٹ بنانے کے لیے تمام ضروریات کی منطقی ترتیب سے فہرست بنائی جائے۔

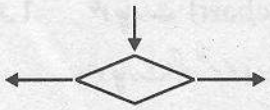
(ii) فلو چارٹ واضح، صاف اور سمجھنے کے لیے آسان ہونا چاہیے۔ فلو چارٹ کو سمجھنے میں کسی بھی کی گنجائش نہیں ہونی چاہیے۔

(iii) کسی طریقہ کار یا سسٹم کے عمومی فلو کی سمت اوپر سے نیچے یا بائیں سے دائیں ہوتی ہے۔

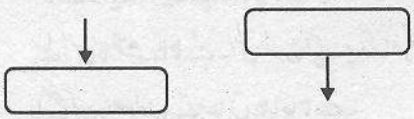
(iv) ایک پروسیس سمبل سے صرف ایک لائن باہر آنی چاہیے۔



(v) فیصلہ کی علامت میں صرف ایک فلولائن داخل ہونی چاہیے، لیکن اس سے ہر ممکن جواب کے لیے ایک اور کل دو لائنیں نکلنی چاہئیں۔

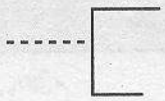


(vi) ٹرمینل (Terminal) سمبل کے ساتھ صرف ایک فلولائن استعمال ہوتی ہے۔



اس بات کا یقین کرلیں کہ فلوچارٹ کا آغاز اور اختتام منطقی شکل میں ہونا چاہیے۔

(vii) ریمارکس کی علامت میں کمینٹس (Comments) تحریر کریں۔ ہم کمینٹس (اینوٹیشن-Annotation) کی علامت مراحل کو مزید واضح بیان کرنے کے لیے استعمال کر سکتے ہیں۔



(viii) اگر فلو چارٹ پیچیدہ ہو جائے تو فلو لائنز کی تعداد کو کم کرنے کے لیے کونیکٹر علامت کا استعمال فائدہ مند ہے۔ اسے مزید پُر اثر اور واضح بنانے کے لیے فلولائنز کو ایک دوسرے کو قطع نہ کرنے دیں۔

(ix) فلوچارٹ میں سادہ ٹیسٹ ڈیٹا دے کر اس کے قابلِ عمل ہونے کو ٹیسٹ کرنا مفید ہے۔

1.4.3 فلوچارٹس کے استعمال کے فوائد (Advantages of using Flowcharts)

فلوچارٹس کے فوائد درج ذیل ہیں۔

1۔ فلوچارٹس کی مدد سے الگورتھم کا حقیقی مفہوم زیادہ مؤثر انداز سے بیان کیا جاسکتا ہے۔

2۔ چونکہ فلو چارٹس ڈیزائن پر مبنی دستاویز کا حصہ ہوتے ہیں، اس لیے اوپریشنل پروگرامز (Operational programmes) کی دیکھ بھال آسان ہوجاتی ہے۔

3۔ پروگرام کی تیاری میں فلوچارٹس گائیڈ (Guide) کی طرح کام آتے ہیں۔ وہ پروگرامر کو پسِ پردہ مسئلہ کے حل کے لیے زیادہ بہتر طور پر کوشش کرنے میں مدد دیتے ہیں۔

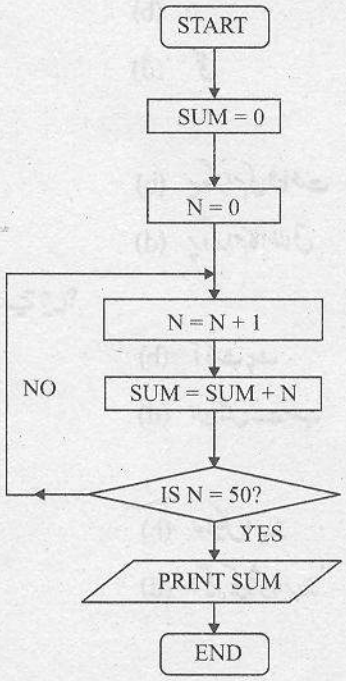
4۔ فلوچارٹ ایررز درست کرنے میں مدد دیتا ہے۔

1.4.4 فلوچارٹس کے استعمال کی حد بندیاں (Limitations of using Flowchasts)

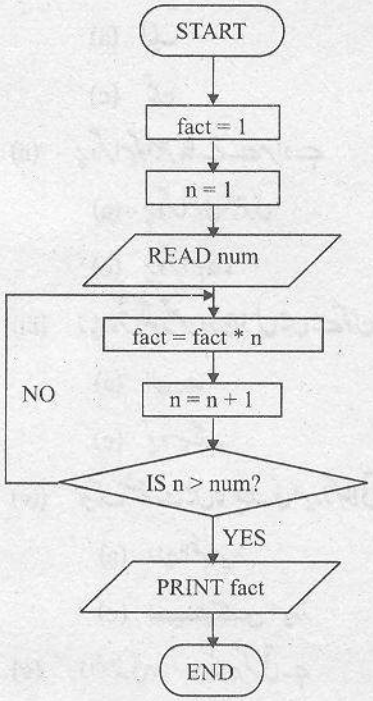
1۔ پیچیدہ مسائل کے لیے فلوچارٹ بنانا مشکل ہوتا ہے۔

2۔ اگر تبدیلیوں کی ضرورت ہو تو فلوچارٹ نئے سرے سے بنانا پڑتا ہے۔

مثال 3: پہلے 50 قدرتی اعداد کا مجموعہ معلوم کرنے کے لیے فلوچارٹ بنائیے۔



مثال 4: کسی نمبر کے فیکٹوریئل کو معلوم کرنے کے لیے فلوچارٹ بنائیے۔



مشق

1- خالی جگہیں پُر کریں۔

(i) کوئی مسئلہ حل کرنے کے لیے کمپیوٹر کو دی گئی ہدایات کا سیٹ ------ کہلاتا ہے۔

(ii) کسی پروگرامنگ لینگویج میں پروگرام لکھنے کے لیے قواعد کا سیٹ اس لینگویج کا ------ کہلاتا ہے۔

(iii) فلوچارٹ الگورتھم کا ------ اظہار ہے۔

(iv) الگورتھم ------ تعداد کے مراحل میں کوئی مسئلہ حل کرتا ہے۔

(v) ------ کے دوران ایک مسئلہ کو کئی چھوٹے مسائل میں تقسیم کیا جاتا ہے۔

(vi) ڈی بگنگ کسی پروگرام میں ------ تلاش کرنے اور دور کرنے کا عمل ہے۔

(vii) پروگرام کے ------ سے مراد استعمال کنندہ کے ماحول میں پروگرام کو انسٹال کرنا ہے۔

(viii) ------ کی سے پروگرام خراب ہو جاتا ہے۔

2- درست جواب منتخب کریں۔

(i) ایک مسئلہ کے کتنے ممکن حل ہیں؟

(a) ایک (b) دو

(c) تین (d) کئی

(ii) پروگرام کو بہتر بنانے سے مراد ہے

(a) پروگرام کی بہتری (b) پروگرام کی شناخت

(c) پروگرام بنانا (d) پروگرام کا اطلاق

(iii) زیادہ تر الگورتھمز درج ذیل میں سے کون سے کام سرانجام دیتے ہیں؟

(a) ان پٹ (b) آؤٹ پٹ

(c) پروسیسنگ (d) ان میں سے سب

(iv) بیسک سٹیٹمنٹ میں ٹائپنگ کی ایررز کہلاتی ہیں

(a) رن ٹائم ایررز (b) لوجیکل ایررز

(c) سینٹیکس ایررز (d) ایگزیکیوشن ایررز

(v) ڈائمنڈ کی علامت ظاہر کرتی ہے

(a) ان پٹ/آؤٹ پٹ (b) فیصلہ سازی

(c) پروسیسنگ (d) ریمارکس

(vi) صفر سے تقسیم ہے

(a) سینٹیکس ایرر (b) لوجیکل ایرر

(c) رن ٹائم ایرر (d) کوئی ایرر نہیں

(vii) ان میں سے کون سی دستاویزات سوفٹ ویئر کی مختلف خصوصیات کو اور ان کے استعمال کو بیان کرتی ہیں؟

(a) سوفٹ ویئر کی ضرورت کا بیان (b) مسئلہ کا ذکر

(c) یوزر مینوئل (d) الگورتھم

(viii) الگورتھم ہے ایک

(a) ضرورت کے بارے میں ڈاکیومنٹ (b) ڈیزائن ڈاکیومنٹ

(c) ٹیسٹ ڈاکومنٹ (d) یوزر گائیڈ

(ix) تقسیم کرو اور فتح کرو کی ٹیکنیک استعمال کرتے ہیں حل کرنے کے لیے

(a) سادہ مسائل (b) پیچیدہ مسائل

(c) بڑے مسائل (d) پیچیدہ اور بڑے مسائل

3- صحیح/غلط کی نشاندہی کریں۔

(i) بیسک، کاروباری ضروریات کو پورا کرنے کی غرض سے بنائی گئی۔

(ii) سینٹیکس ایررز پروگرام کے غلط لوجک کی وجہ سے ہوتی ہیں۔

(iii) پیچیدہ مسائل حل کرنے کے لیے ٹاپ ڈاؤن ڈیزائن کی پیروی کی جاتی ہے۔

(iv) الگورتھم کے کام کی تصدیق کا عمل ڈیسک چیکنگ کہلاتا ہے۔

(v) ڈی بگنگ تجزیہ کا اہم حصہ ہے۔

(vi) پروگرام سازی کا ہر مرحلہ تحریری شکل میں محفوظ کیا جاتا ہے۔

(vii) فلوچارٹ میں مستطیل کی علامت فیصلہ سازی کے لیے استعمال ہوتی ہے۔

(viii) ڈویلپمنٹ کے مرحلہ میں ضرورت سے متعلقہ ڈاکیومنٹ مددگار ثابت نہیں ہوتا۔

(ix) فلوچارٹ کی عمومی سمت دائیں سے بائیں طرف ہوتی ہے۔

(x) اینوٹیشن کی علامت کمینٹس تحریر کرنے کے لیے استعمال کی جاتی ہے۔

4- مسئلہ کے حل سے کیا مُراد ہے؟ مسئلوں کے حل کا پروسیس مختصر طور پر بیان کیجیے۔

5- ڈی بگنگ کیا ہے؟ پروگرام میں کتنی قسم کی ایررز واقع ہوسکتی ہیں؟ مختصر طور پر وضاحت کیجیے۔

6- الگورتھم کی تعریف بیان کیجیے۔ اپنے دوست کو ٹیلی فون کال کرنے کے لیے مرحلہ وار الگورتھم لکھیے۔

7- فلوچارٹ کے کیا فوائد ہیں؟ فلوچارٹس کی حد بندیاں بیان کیجیے۔

8- تین اعداد میں سے سب سے بڑا عدد معلوم کرنے کے لیے فلوچارٹ بنائیے۔

9- دائرہ کا رقبہ معلوم کرنے کے لیے الگورتھم لکھیے، جبکہ نصف قطر دیا گیا ہو۔

(اشارہ: رقبہ = 3.14*r*r)

10- درج ذیل سوالات کے مختصر جواب دیجیے۔

(i) مسئلہ حل کرنے کے مراحل کی فہرست تحریر کریں۔

(ii) تجزیہ سے کیا مراد ہے؟ مسئلہ کے حل میں اس کی اہمیت بیان کریں۔

(iii) پیچیدہ مسائل حل کرنے کے لیے کون سا طریقہ اختیار کرنا چاہیے؟ مختصر طور پر بیان کریں۔

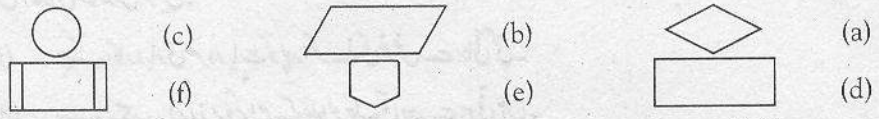
(iv) پروگرامنگ لینگوئج کے سینٹیکس سے آپ کیا مراد لیتے ہیں؟ کیا کمپیوٹر پر مسئلہ حل کرنے کے لیے سینٹیکس کا جاننا ضروری ہے؟

(v) رن ٹائم ایررز اور لوجیکل ایررز میں فرق بیان کریں۔

(vi) مسئلہ حل کرنے کے پروسیس میں دستاویز سازی کا عمل کیوں اہم تصور کیا جاتا ہے؟ وجوہات بیان کریں۔

(vii) کیا کوئی مسئلہ حل کرنے کے لیے الگورتھم کے مراحل کا محدود تعداد میں ہونا ضروری ہے؟ اگر ہاں تو کیوں؟

(viii) فلوچارٹ کی مندرجہ ذیل علامات کا مطلب بیان کریں۔



(a) (b) (c)

(d) (e) (f)

(ix) فلوچارٹ اور الگورتھم کا موازنہ کریں۔

(x) v ms^{-1} کی اوسط رفتار سے حرکت کرتی ہوئی کار کا t وقت میں طے کردہ فاصلہ شمار کرنے کے لیے الگورتھم لکھیں۔ پروگرام کو چاہیے کہ اوسط رفتار v اور وقت t کی قیمتیں لے۔ (اشارہ: $s = vt$ جبکہ s سے مراد ہے طے کردہ فاصلہ)

جوابات

-1

(i) پروگرام (ii) سینٹیکس (iii) تصویری

(iv) واضح (v) تجزیہ (vi) ایررز

(vii) عملی اطلاق (viii) پروگرام چلنے کے دوران

-2

(i)	d	(ii)	a	(iii)	d	(iv)	c	(v)	b
(vi)	c	(vii)	c	(viii)	b	(ix)	d		

-3

(i)	غلط	(ii)	غلط	(iii)	صحیح	(iv)	صحیح	(v)	غلط
(vi)	صحیح	(vii)	غلط	(viii)	غلط	(ix)	غلط	(x)	صحیح

باب 2

ڈیٹا ٹائپس، اسائنمنٹ اور ان پٹ/آؤٹ پٹ سٹیٹمنٹس

(Data Types, Assignment and Input/Output Statements)

2.1 تعارف (Introduction)

بیسک لینگوئج (Basic Language) کو 1963ء میں جان کیمنی (John Kemeny) اور تھامس کرٹز (Thomas Kurtz) نے ڈارماؤتھ کالج، امریکہ میں ایجاد کیا۔ اسے ہدایاتی آلہ کے طور پر طلبا کو پروگرامنگ کے بنیادی تصورات سکھانے اور پرانی لینگوئجز (Languages) کے پیچیدہ امور طے کرنے کے لیے ایجاد کیا گیا۔

جی ڈبلیو۔ بیسک لینگوئج کا انٹرپریٹر (Interpreter) ہے۔ بیسک لینگوئج کے بہت سے دوسرے ٹرانسلیٹرز ہیں جو بیسک لینگوئج فروخت کرنے والے مختلف اداروں نے پیش کیے ہیں جیسا کہ کیو بیسک (QBASIC) یعنی کوئیک (Quick) بیسک جس میں پروگرام لکھنے اور چلانے کے لیے مینیوز (menus) کی شکل میں کمانڈز (Commands) پیش کی جاتی ہیں۔ یہاں ہم صرف جی ڈبلیو۔ بیسک کا حوالہ دیں گے کیونکہ یہ استعمال میں سادہ اور آسان ہے۔

2.2 اوپریشن کے طریقہ کار (Modes of Operation)

جی ڈبلیو۔ بیسک سے دو طریقوں سے کام لیا جا سکتا ہے یعنی ڈائریکٹ موڈ (Direct mode) سے اور ان ڈائریکٹ موڈ (Indirect mode) سے۔ جب جی ڈبلیو۔ بیسک پروگرام لوڈ کیا جاتا ہے تو یہ Ok کا پیغام دیتا ہے۔ اس مرحلہ پر، یہ ڈائریکٹ موڈ میں ہوتا ہے۔

اِن ڈائریکٹ موڈ میں، جی ڈبلیو۔ بیسک کی کمانڈ ٹائپ کرتے ہی ایگزیکیوٹ ہو جاتی ہیں۔ ارتھمیٹک (Arithmetic) اور لوجیکل (Logical operations) کے نتائج فوراً دیکھے جا سکتے ہیں، لیکن ایگزیکیوشن (Execution) کے بعد کمانڈز ختم ہو جاتی ہیں۔ یہ موڈ ایرر درست کرنے اور تیزی سے ایسے حساب کرنے کے لیے کارآمد ہے جس کے لیے مکمل پروگرام کی ضرورت نہیں ہوتی۔

```
Ok
PRINT 786/3
  262
Ok

PRINT   " Welcome to GW-BASIC "
Welcome to GW-BASIC
Ok
```

شکل 2.1: ڈائریکٹ موڈ کی مثال

اِن ڈائریکٹ موڈ پروگرام ٹائپ کرنے کے لیے استعمال ہوتا ہے۔ پروگرام سٹیٹمنٹس کے شروع میں لائن نمبر دیے جاتے ہیں اور یہ میموری (Memory) میں سٹور ہو جاتی ہیں۔ میموری میں لوڈ (Load) کیا ہوا پروگرام RUN کی کمانڈ سے ایگزیکیوٹ کیا جاتا ہے۔

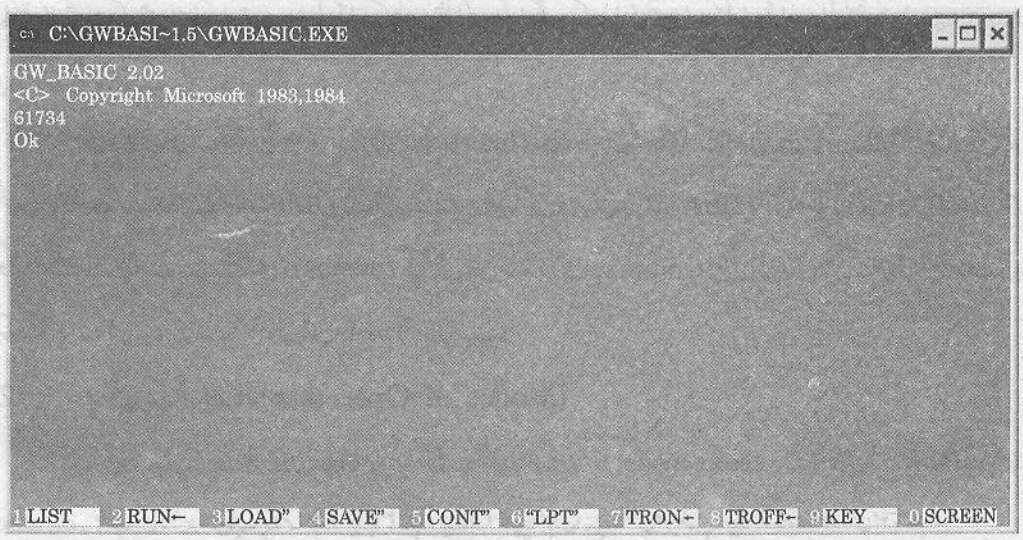
```
Ok
auto
10 PRINT 786/3
20 PRINT "Welcome to GW-BASIC"
30 END
Ok

RUN
 262
Welcome to GW-BASIC
Ok
```

شکل 2.2: اِن ڈائریکٹ موڈ کی مثال

2.3 جی ڈبلیو-بیسک میں پروگرامز لکھنا (Writing Programs in GWBASIC)

جی ڈبلیو-بیسک پروگرام لکھنے کے لیے ایک ایڈیٹر (Editor) فراہم کرتی ہے۔ یہ حقیقت میں صرف ایک ایڈیٹر ہی نہیں ہوتا بلکہ پروگرام ڈویلپمنٹ کا مکمل ماحول (Integrated Development Environment) ہوتا ہے جسے مختصر طور پر IDE کہتے ہیں۔ اس میں ہم بیسک پروگرامز لکھ سکتے ہیں، ایررز درست کر سکتے ہیں، پروگرام محفوظ کر سکتے ہیں، میموری میں لوڈ کر سکتے ہیں اور چلا سکتے ہیں (شکل 2.3)۔



شکل 2.3: جی ڈبلیو-بیسک IDE

2.3.1 پروگرام بنانا اور محفوظ کرنا (Creat and Save the Program)

پروگرام ایک فائل (File) ہوتی ہے جس میں کمپیوٹر کے لیے مخصوص ہدایات یا بیانات شامل ہوتے ہیں۔ جی ڈبلیو-بیسک پروگرام میں لائنوں کا مندرجہ ذیل فارمیٹ ہوتا ہے۔

Line # statement(s)

جہاں لائن # صفر (0) سے 65529 کی رینج (Range) تک ایک عدد ہوتا ہے۔ سٹیٹمنٹ سے مراد جی ڈبلیو-بیسک کی کوئی سٹیٹمنٹ ہے۔ جی ڈبلیو-بیسک پروگرام لائن ہمیشہ ایک لائن نمبر سے شروع ہوتی ہے اور اس میں کم از کم ایک کریکٹر (Character) ہوتا ہے مگر 255 سے زیادہ کریکٹرز نہیں ہو سکتے۔ تاہم، لائن پر ایک سے زیادہ سٹیٹمنٹس ہو سکتی ہیں۔ اگر ایسا ہو تو ہر ایک سٹیٹمنٹ کو کولن (:) کے ذریعے الگ کیا جاتا ہے۔ پروگرام کی سٹیٹمنٹس لائن نمبر پر ترتیب صعودی کے لحاظ سے ایگزیکیوٹ کی جاتی ہیں۔ مثلاً اگر آپ لائن نمبر 45 کو لائن نمبر 60 کے بعد بھی انٹر کریں تو کمپیوٹر پھر بھی لائن نمبر 45 کو لائن نمبر 60 سے پہلے چلائے گا۔

موجودہ لائن نمبر کا دوبارہ استعمال' اصل لائن میں درج تمام معلومات کے گم ہونے کا باعث بنتا ہے۔ اس طرح آپ سے حادثاتی طور پر کچھ پروگرام لائنز ختم ہوسکتی ہیں۔

پروگرام کو آئندہ استعمال کرنے کے لیے ہمیں اسے لازمی طور پر محفوظ (سیو - Save) کرنا چاہیے۔ جی ڈبلیو - بیسک میں فائل محفوظ کرنے کے لیے مندرجہ ذیل طریقہ کار استعمال کیا جاتا ہے۔

1۔ F4 بٹن کو دبائیں یا SAVE کی کمانڈ ٹائپ کریں۔

2۔ ڈبل کیوٹیشنز (" ") میں پروگرام کا نام ٹائپ کریں اور اینٹر - کی (Enter-key)/ ریٹرن - کی (Return-key) دبائیں۔ فائل آپ کے مخصوص کردہ نام کے تحت محفوظ ہوجائے گی۔

2.3.2 پروگرام لوڈ کرنا (Load the Program)

پروگرام کو لوڈ کرنے کا مطلب اس کو سیکنڈری سٹوریج (Secondary storage) آلہ (جیسا کہ ہارڈ ڈسک Hard disk) سے میموری میں لانا ہے تاکہ اس کی ہدایات پر عمل کیا جاسکے۔ ایک محفوظ کیا ہوا پروگرام مندرجہ ذیل طریقہ سے لوڈ کیا جاسکتا ہے۔

1۔ F3 بٹن دبائیں یا LOAD کی کمانڈ ٹائپ کریں۔

2۔ ڈبل کیوٹیشنز (" ") میں فائل کا نام ٹائپ کریں۔ 3۔ اینٹر - کی کا بٹن دبائیں۔

فائل' میموری میں لوڈ کی جاتی ہے اور فہرست دیکھنے' ایڈٹ کرنے اور چلنے کے لیے تیار ہوجاتی ہے۔

☆ اگر فائل موجود نہ ہو یا اس کا پاتھ غلط ٹائپ ہوجائے تو ایرر کا پیغام نظر آتا ہے۔

☆ جی۔ ڈبلیو بیسک پروگرام کی موجودہ ایکسٹینشن .bas ہوتی ہے۔

☆ کسی ڈائریکٹری (Directory) میں فائل محفوظ کرنے کے لیے فائل کے نام کے ساتھ اس کا مکمل ایڈریس (پاتھ) ٹائپ کرنا چاہیے۔ بصورتِ دیگر فائل جی ڈبلیو - بیسک کی موجودہ ورکنگ ڈائریکٹری میں محفوظ ہوجائے گی۔

2.3.3 پروگرام چلانا (Execute the Program)

پروگرام چلانے سے مراد پروگرام میں ہدایات پر عمل کرنا ہے۔ پروگرام چلانے سے پہلے اسے میموری میں لوڈ کرنا چاہیے۔ اس لیے ایک پروگرام چلانے کے لیے' پہلے اُسے اُوپر بتائے گئے طریقہ سے لوڈ کریں۔ پھر F2 بٹن دبائیں یا RUN کی کمانڈ ٹائپ کریں۔ اس کے نتیجہ میں سکرین پر پروگرام کا آؤٹ پُٹ ظاہر ہوگا۔

```
Ok
10 LET x = 10: LET y = 20
20 PRINT "SUM = "; x + y
30 END
SAVE"C:\sum. bas"
Ok
```

شکل 2.4: پروگرام بنانا اور محفوظ کرنا

```
Ok
LOAD "c:\sum.bas"

RUN
SUM =  30
Ok
```

شکل 2.5: پروگرام لوڈ کرنا اور چلانا

2.4 بیسک پروگرام کا ڈھانچہ (Structure of the BASIC Program)

بیسک پروگرام تحریر کرتے وقت مندرجہ ذیل اصولوں کی پابندی کرنی چاہیے۔

1۔ ہر پروگرام سٹیٹمنٹ ایک لائن نمبر سے شروع ہونی چاہیے۔

2۔ ہر بیسک پروگرام کو END سٹیٹمنٹ سے ختم کرنا اچھی بات ہے۔ تاہم یہ ضروری نہیں ہے۔

3۔ ایک پروگرام کے اندر لائن نمبرز کی تکرار کی اجازت نہیں ہے۔

4۔ دو یا دو سے زیادہ سٹیٹمنٹس کو ایک ہی لائن پر لکھا جا سکتا ہے تاہم ان کو ایک کولن (:) سے علیٰحدہ کرنا ضروری ہے۔

5۔ بیسک میں متغیرات بغیر بتلائے استعمال کیے جا سکتے ہیں۔

6۔ BASIC پروگرام میں پروگرام سٹیٹمنٹس کی ترتیب ضروری نہیں ہے۔ مثال کے طور پر، آپ لائن # 90 کو لائن # 60 سے پہلے لکھ سکتے ہیں؛ تاہم پروگرام سٹیٹمنٹس پر ہمیشہ لائن نمبرز کے مطابق عمل کیا جاتا ہے۔ اس لحاظ سے لائن #60 پر پہلے عمل ہوگا اور لائن #90 پر بعد میں عمل ہوگا خواہ ان کے لکھنے کی ترتیب کچھ بھی ہو۔

2.5 BASIC کے کریکٹر سیٹ (Character set of BASIC)

کسی بھی لینگوئج کے کریکٹر سیٹ سے مراد وہ تمام کریکٹرز ہیں جو کہ اُس لینگوئج میں پروگرام لکھنے کے لیے استعمال ہو سکتے ہیں۔ GW-BASIC کا کریکٹر سیٹ ایلفابیٹس (Alphabets)، نومیرک (Numeric) اور کُچھ خاص کریکٹرز پر مشتمل ہوتا ہے۔

☆ ایلفابیٹس میں بڑے اور چھوٹے حروف شامل ہیں۔

☆ نومیرک کریکٹرز میں 0 سے 9 تک ہندسے شامل ہوتے ہیں۔

☆ GW-BASIC کریکٹر سیٹ میں درج ذیل خاص کریکٹرز شامل ہوتے ہیں۔

کریکٹرز	مفہوم
BLANK	
=	مساوی یا اسائنمنٹ (Assignment) علامت
+	جمع کی علامت یا سٹرنگ (String) کریکٹرز کو ملانے کی علامت
-	تفریق کی علامت
*	ضرب کی علامت
/	تقسیم کی علامت
^	قوت نما یا کنٹرول (Control) کی علامت
(	بائیں بریکٹ
)	دائیں بریکٹ

%	فیصد کی علامت
#	نمبر کی علامت
$	ڈالر کی علامت یا سٹرنگ ویری ایبل (Variable) کی علامت
!	استعجابیہ (Exclamation) علامت
[	بائیں بڑی بریکٹ
]	دائیں بڑی بریکٹ
,	کوما
""	ڈبل کوٹیشن کی علامت
.	وقفہ، نقطہ یا اعشاریہ کی علامت
'	سنگل کوٹیشن کی علامت، اپوسٹرافی (Apostrophe)، یا ریمارک کی علامت
;	سیمی کولن یا اینٹر۔ کی دبانے کی علامت
:	کولن یا لائن سٹیٹمنٹ کے ساتھ استعمال ہونے والی علامت
&	اینڈ (اور) کی علامت یا ہیکسا ڈیسیمل اور آکٹل نمبر کے ساتھ استعمال ہونے والی علامت
?	سوالیہ علامت
<	چھوٹا ہونے کی علامت، مقابلتاً کم کی علامت
>	بڑا ہونے کی علامت، مقابلتاً زیادہ کی علامت
\	بیک سلیش (Back slash) کی علامت، واپسی کی علامت یا مکمل عدد کو تقسیم کرنے کی علامت
@	ایٹ (At) کی علامت
_	انڈر سکور (Underscore)
BACKSPACE	بیک سپیس، آخری ٹائپ کیا ہوا حرف مٹاتی ہے
ESC	سکرین سے موجودہ لائن مٹاتی ہے
TAB	اگلے ٹیب (Tab) سٹاپ پر لے جاتی ہے
RETURN	لائن ختم کرتی ہے اور کرسر (Cursor) اگلی لائن کے شروع میں لے جاتی ہے

2.6 ذخیرہ الفاظ (Reserved Words)

ذخیرہ الفاظ یا کی ورڈز (Keywords) ایسے الفاظ ہیں جن کا مطلب بیسک میں پہلے ہی بیان کر دیا گیا ہے۔ ان کا استعمال پہلے سے ہی

طے شدہ ہے اور یہ پروگرام میں کسی اور مقصد کے لیے استعمال نہیں ہو سکتے اور نہ ہی ان کے استعمال کا مقصد تبدیل کیا جا سکتا ہے۔ کی ورڈز کو متغیرات کے ناموں کے طور پر استعمال نہیں کیا جا سکتا۔ بیسک کے کچھ کی ورڈز IF, ELSE, THEN, WHILE وغیرہ ہیں۔

2.7 متغیرات (Variables)

متغیرات کو میموری لوکیشنز (Memory locations) / میموری سیلز (Memory cells) کا نام دیا جاتا ہے جن کو پروگرام کے ان پُٹ ڈیٹا کو محفوظ کرنے کے لیے اور پروگرام کی ایگزیکیوشن کے دوران کمپیوٹیشنل نتائج کو ذخیرہ کرنے کے لیے استعمال کیا جاتا ہے۔

جیسا کہ نام سے ظاہر ہے، متغیر کی قیمت پروگرام کی ایگزیکیوشن کے دوران تبدیل ہو سکتی ہے۔ اگر متغیر کو کوئی قیمت نہ دی جائے تو GW-BASIC عددی متغیر کی قیمت صفر مقرر کرتا ہے اور سٹرنگ متغیر کو کوئی قیمت نہیں دیتا۔

2.7.1 بیسک میں متغیرات کے ناموں کے قوانین (Rules For Naming Variables in BASIC)

بیسک پروگرام میں استعمال ہونے والے ہر متغیر کا ایک نام ہونا چاہیے۔ متغیر کا نام اس سے متعلق مزید حوالوں کے لیے استعمال ہوتا ہے۔ متغیر کی قیمت تک رسائی اس کے نام سے ہوتی ہے۔ GW-BASIC میں متغیرات کے ناموں کے لیے کچھ اصول وضع کیے گئے ہیں جو کہ یہ ہیں۔

1۔ GW-BASIC میں متغیر کا نام 40 کریکٹرز سے زیادہ لمبا نہیں ہو سکتا۔

2۔ متغیر کا نام ایلفابیٹس (بڑے اور چھوٹے حروف)، اعداد اور اعشاری پوائنٹ پر مشتمل ہو سکتا ہے۔

3۔ متغیر کے نام کا پہلا کریکٹر ایک ایلفابیٹ ہونا چاہیے۔

4۔ ذخیرہ الفاظ (reserved words) کو متغیر کے ناموں کے لیے استعمال نہیں کیا جا سکتا۔

5۔ متغیر کے ناموں کے درمیان خالی جگہوں کی اجازت نہیں ہے۔

6۔ تاہم، کسی متغیر کا آخری کریکٹر متغیر کی قسم کو ظاہر کرنے والے خاص کریکٹر پر مشتمل ہو سکتا ہے۔

اگر کسی متغیر کی قسم بیان نہ کی جائے تو اسے حقیقی قسم (real type) کا متغیر تصور کیا جاتا ہے۔

2.7.2 ٹائپ ڈیکلیریشن کریکٹرز (Type Declaration Characters)

GW-BASIC میں ٹائپ ڈیکلیریشن کریکٹرز متغیر کی قسم کو ظاہر کرتے ہیں۔ اس میں درج ذیل ٹائپ ڈیکلیریشن کریکٹرز تسلیم شدہ ہیں۔

کریکٹر	متغیر کی قسم	مثال	مطلوب میموری
$	String variable	Name $	String length
%	Integer variable	Marks %	2 Bytes
!	Single-Precision variable	Avg !	4 Bytes
#	Double-precision variable	Area #	8 Bytes

2.7.3 متغیرات کی اقسام (Types of Variables)

متغیرات کی دو بنیادی اقسام ہیں۔

☆ نومیرک متغیرات ☆ سٹرنگ متغیرات، جو کریکٹرز کے سٹرنگز کو ذخیرہ کر سکتے ہیں

نومیرک متغیرات (Numeric Variables)

نومیرک متغیرات نومیرک قیمتوں کو ذخیرہ کر سکتے ہیں۔ (نومیرک قیمتوں میں فلوٹنگ پوائنٹ (Floating point) نمبرز اور مکمل اعداد دونوں شامل ہیں) اگر ہم نومیرک متغیر کی قسم بیان نہیں کرتے تو جی ڈبلیو بیسک اسے ایک لفظی (single-precision) متغیر تصور کرتا ہے۔ ایک لفظی متغیرات چھ اہم ہندسوں تک اعداد کو صحیح طور پر ہینڈل کر سکتے ہیں۔ تاہم یہ ساتویں اہم ہندسے کو صحیح طور پر ہینڈل (Handle) نہیں کر سکتے۔ اگر آپ مزید درستگی چاہتے ہیں تو آپ کو دو لفظی (double-precision) فارمیٹ استعمال کرنا چاہیے۔

سٹرنگ متغیرات (String Variables)

ایک سٹرنگ سے مراد ڈبل کوٹیشن میں بند ترتیب وار کریکٹرز ہیں۔ ایک سٹرنگ متغیر ترتیب وار کریکٹرز کو سٹور کر سکتا ہے۔ نوعیت کے لحاظ سے کریکٹر سٹرنگ، نومیرک سٹرنگ سے بالکل مختلف ہوتا ہے۔ بیسک اور دوسری لینگوئجز نومیرک اور سٹرنگ ڈیٹا کو محفوظ کرنے کے لیے مختلف فارمیٹ استعمال کرتی ہیں۔ بیسک میں سٹرنگ متغیر کے نام کے بعد ڈالر کی علامت ($) استعمال ہوتی ہے۔ ہم نومیرک قیمتوں پر سرانجام دیے جانے والے عوامل سٹرنگز پر سرانجام نہیں دے سکتے۔ مثال کے طور پر سٹرنگ کو جمع، تفریق، ضرب اور تقسیم نہیں کیا جا سکتا۔ سٹرنگز پر کچھ دیگر عوامل سرانجام دیے جا سکتے ہیں جیسا کہ دو قیمتوں کو ملانا اور موازنہ کرنا۔

انٹیجر، ایک لفظی اور دو لفظی متغیرات کی باہمی تبدیلی کے وقت بہت محتاط رہیے۔ ایررز (Errors) کی صورت میں درستگی کے امکانات معدوم ہو جاتے ہیں۔

2.8 کانسٹینٹ (Constant)

کانسٹینٹ ایک ایسی مقدار ہے جس کی قیمت تبدیل نہیں ہو سکتی۔ پروگرام کی ایگزیکیوشن کے دوران متغیر کی طرح کانسٹینٹ کی قیمت تبدیل نہیں ہو سکتی۔ بیسک میں دو طرح کے کانسٹینٹس ہیں۔ نومیرک کانسٹینٹ اور سٹرنگ کانسٹینٹ ہیں۔

2.8.1 نومیرک کانسٹینٹس (Numeric Constants)

نومیرک کانسٹنٹ انٹیجر، ایک لفظی یا دو لفظی اعداد پر مشتمل ہوتے ہیں۔ انٹیجر کانسٹینٹ ایسی قیمتوں کو ظاہر کرتے ہیں جنہیں گنا جاتا ہے اور ان میں کسری حصہ نہیں ہوتا۔ مثال کے طور پر 8 ,-578 ,56+ وغیرہ۔ ایک لفظی اور دو لفظی کانسٹینٹس ایسی قیمتوں کو ظاہر کرتے ہیں جنہیں گنا جا سکتا ہے اور ان میں کسری حصہ شامل ہو سکتا ہے۔ مثال کے طور پر 0.45 ,5.0 ,-4.786 وغیرہ۔ ایک لفظی نومیرک کانسٹینٹس کو 7 ہندسوں سے سٹور کیا جاتا ہے (اگرچہ صرف 6 ہندسے درست ہو سکتے ہیں)۔ دو لفظی نومیرک کانسٹینٹس کو 17 ہندسوں سے سٹور کیا جاتا ہے اور زیادہ سے زیادہ 16 ہندسوں سے پرنٹ کیا جاتا ہے۔

ایک لفظی کانسٹینٹ مندرجہ ذیل میں سے کوئی بھی نومیرک کانسٹینٹ ہو سکتا ہے۔

☆ سات یا کم ہندسوں کے ساتھ ☆ قوت نمائی انداز میں E استعمال کرتے ہوئے

☆ بعد میں آنے والی استعجابیہ (Exclamation) علامت کے ساتھ ایک دولفظی کانسٹینٹ مندرجہ ذیل میں سے کوئی بھی نومیرک کانسٹینٹ ہوسکتا ہے۔

☆ آٹھ یا زیادہ ہندسوں کے ساتھ ☆ قوّت نمائی انداز میں D استعمال کرتے ہوئے

☆ ایک عددی نشان (#) کے ساتھ ایک اور دولفظی نومیرک کانسٹینٹس کی مثالیں مندرجہ ذیل ہیں۔

ایک لفظی کانسٹینٹس	دولفظی کانسٹینٹس
23.08	342237861
-3.15E-04	-5.35857D-06
2145.0	3220.0#
37.4!	7645721.1334

2.8.2 سٹرنگ کانسٹینٹس (String Constants)

ایک سٹرنگ کانسٹینٹ ڈبل کوٹیشن مارکس میں بند ترتیب وار نومیرک کریکٹرز پر مشتمل ہوتا ہے۔ سٹرنگ کانسٹینٹ کی زیادہ سے زیادہ لمبائی 255 کریکٹرز ہوتی ہے۔ مثال کے طور پر "Lahore", "4900", "I Love Pakistan" وغیرہ۔

2.9 بیسک کمانڈز (BASIC Commands)

بیسک لینگوئج میں عام استعمال ہونے والی چند کمانڈز درج ذیل ہیں۔

2.9.1 AUTO Command

یہ کمانڈ RETURN کی کو دبانے پر خود بخود لائن نمبرز کو صعودی ترتیب میں جنریٹ (Generate) کرتی ہے۔

سینٹیکس Syntax

AUTO [line number] [,[increment]]

AUTO.[,[increment]]

وضاحت (Interpretation)

AUTO کی کمانڈ پروگرام ٹائپ کرنے کے لیے کارآمد ہے کیونکہ یہ ہمیں غیر ضروری لائن نمبرز ٹائپ کرنے سے آزاد کرتی ہے۔ اُوپر بیان کیے گئے سینٹیکس میں [line number] 'پروگرام کی پہلی لائن نمبر بیان کرتا ہے جبکہ [increment] 'پچھلی لائن نمبر میں مخصوص اضافہ ظاہر کرتا ہے۔ دونوں کی موجودہ قیمت 10 ہے۔

پیریڈ (Period) (.) کو لائن نمبر کے نعم البدل کے طور پر موجودہ لائن کی نشاندہی کرنے کے لیے استعمال کیا جا سکتا ہے۔ اگر لائن نمبر کے بعد [increment] کو نہ دیا جائے تو پہلے دی گئی AUTO کمانڈ میں دیا گیا انکریمنٹ (Increment) فرض کر لیا جاتا ہے۔

اگر AUTO 'لائن نمبر کو جنریٹ کرے جو کہ پہلے بھی استعمال کی جا رہی ہو تو باخبر کرنے کے لیے عدد کے بعد ایک * ظاہر ہوتا ہے کہ کسی ان پٹ سے موجودہ لائن ختم ہو جائے گی۔ تاہم ' * کے فوراً بعد ریٹرن دبانے سے لائن محفوظ ہو جاتی ہے اور اگلا لائن نمبر

جنریٹ ہو جاتا ہے۔AUTO کو CTRL-BREAK یا CTRL-C اینٹر کرتے ہوئے ختم کیا جاتا ہے۔ پھر جی ڈبلیو۔بیسک کمانڈ لیول (Command level) کی طرف واپس چلی جاتی ہے۔

مثالیں:

AUTO 100, 200

یہ لائن نمبر 100، 120، 140 کو اور اسی ترتیب سے بعد والی لائنوں کو جنریٹ کرتی ہے۔

AUTO

یہ لائن نمبر 10، 20، 30، 40 کو اور اسی ترتیب سے بعد والی لائنوں کو جنریٹ کرتی ہے۔

AUTO, 50

یہ آخری AUTO کمانڈ میں دیے گئے لائن نمبر میں 50 کے اضافہ کے ساتھ لائن نمبر جنریٹ کرتی ہے۔

AUTO.,

یہ پچھلی اینٹر کی گئی AUTO کمانڈ میں مخصوص اضافہ کے ساتھ موجودہ لائن نمبر سے شروع ہو کر لائن نمبرز کو جنریٹ کرتی ہے۔

2.9.2 CLEAR Command

یہ کمانڈ تمام نومیرک متغیرات کی قیمتوں کو صفر سیٹ کرتی ہے اور تمام سٹرنگ متغیرات کی قیمتوں کو نل (null) اور تمام کھلی ہوئی فائلز کو بند کرتی ہے۔

```
Ok
X = 100
Ok

PRINT X
 100
Ok

CLEAR
Ok

PRINT X
 0
Ok
```

سینٹیکس **Syntax**

CLEAR

مثال: درج ذیل شکل CLEAR کمانڈ کے اثر کو ظاہر کرتی ہے۔

2.9.3 CLS Statement

یہ کمانڈ سکرین کو کلیئر کرنے کے لیے استعمال ہوتی ہے۔

سینٹیکس **Syntax**

CLS [n]

وضاحت (Interpretation)

یہاں n اختیاری ہے لیکن درج ذیل میں سے اس کی قیمت کچھ بھی ہو سکتی ہے۔

n کی قیمت	اثر
0	سکرین سے تمام متن اور گرافکس (Graphics) کو کلیئر (Clear) کرتی ہے۔
1	سکرین سے صرف گرافکس کو کلیئر کرتی ہے۔
2	سکرین سے صرف متن کو کلیئر کرتی ہے۔

مثال: کمانڈ لیول پر CLS ٹائپ کیجیے۔

CLS

CLS 1

DELETE Command 2.9.4

یہ کمانڈ پروگرام لائنز یا لائن رینجز کو ڈیلیٹ (Delete) کرنے کے لیے استعمال ہوتی ہے۔

Syntax **سینٹیکس**

DELETE [line number 1] ┣ line number 2]

DELETE line number 1-

مثالیں:

DELETE 70

یہ کمانڈ لائن نمبر 70 کو ڈیلیٹ کرتی ہے۔

DELETE 50-150

یہ کمانڈ لائن نمبرز 50 تا 150 کو ڈیلیٹ کرتی ہے۔

DELETE - 80

یہ کمانڈ لائن نمبر 80 تک تمام لائنوں کو ڈیلیٹ کرتی ہے۔

DELETE120-

یہ کمانڈ لائن نمبر 120 تا آخری لائن تک تمام لائنوں کو ڈیلیٹ کرتی ہے۔

EDIT Command 2.9.5

یہ کمانڈ پروگرام لائن میں تبدیلی کرنے کے لیے استعمال ہوتی ہے۔

Syntax **سینٹیکس**

EDIT line number

EDIT .

وضاحت (Interpretation)

لائن نمبر سے مراد پروگرام کی اس لائن کا نمبر ہے جسے ہم ایڈٹ کرنا چاہتے ہیں۔ پیریڈ (.) سے مُراد موجودہ لائن ہے۔

مثالیں:

EDIT 140

یہ کمانڈ پروگرام کی لائن نمبر 140 کو ایڈیٹنگ کے لیے ظاہر کرتی ہے۔

EDIT

موجودہ پروگرام لائن کو ایڈیٹنگ کے لیے ظاہر کرتی ہے۔

FILES Command 2.9.6

یہ کمانڈ مخصوص ڈرائیو پر موجود تمام فائلز کے ناموں کی فہرست کے لیے استعمال ہوتی ہے۔

Syntax **سینٹیکس**

FILES [pathname]

وضاحت (Interpretation)

[pathname] اوپشنل پیرامیٹر (Optional Parameter) ہے جو اگر نہ دیا جائے تو کمانڈ منتخب شدہ ڈرائیو کی موجودہ ڈائریکٹری کی تمام فائلوں کی فہرست دکھاتی ہے۔ ہاتھ میں وائلڈ کارڈز جیسا کہ * اور ؟ بھی استعمال کیے جا سکتے ہیں۔ سوالیہ علامت (?) فائل کے نام یا ایکسٹینشن (Extension) میں شامل کسی بھی کریکٹرز کو ملانے جب کہ * کسی بھی فائل کے نام یا ایکسٹینشن کے طور پر استعمال ہوتی ہے۔

مثالیں:

FILES

منتخب ڈرائیو کی موجودہ ڈائریکٹری میں تمام فائلز کی فہرست دکھاتی ہے۔

FILES "*.doc"

doc ایکسٹینشن والی تمام فائلز کی فہرست دکھاتی ہے۔

FILES "D: *.* "

D ڈرائیو پر کسی بھی ایکسٹینشن کے ساتھ تمام فائلز کی فہرست دکھاتی ہے۔

FILES "Mar? .xls"

ان تمام فائلز کی فہرست دکھاتی ہے جن

- کی ایکسٹینشن xls. ہے
- کا نام چار کریکٹرز پر مشتمل ہو۔
- کے ناموں کے پہلے تین کریکٹرز Mar جبکہ چوتھا کریکٹر کچھ بھی ہو سکتا ہے۔

```
Ok
FILES
C:\GWBASI~1.5
GWBASIC .EXE        GWBASIC3.EXE          SMINFO.   SYS
  660643840 Bytes free
Ok

FILES "G:*.*"
G:\
1STYEA~1            1STYEA~2              2NDYEA~1              2NDYEA~2
ADMISS~1    <DIR>   ADNAN       <DIR>     AMJAD       <DIR>     CHAT        <DIR>
CLASS1~1.XLS        CLASS12 .XLS          FINAL2~1    <DIR>     GORIYA~1.RTF
PAPER       <DIR>   PRIMEM~1.DOC          PROGRA~1    <DIR>     RESULT~1.XLS
RESULT~2.XLS        RESULT~3.XLS          TESTRE~1    <DIR>     TOPOLOGY .DOC
XII-CL~1.XLS        XII-CL~2.XLS
  1023932928 Bytes free
Ok
```

شکل نمبر 2.8: FILES کمانڈ کی مثال

KILL Command 2.9.7

یہ کمانڈ کسی فائل کو ڈسک سے ہٹانے یا ختم کرنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

KILL filename

وضاحت (Interpretation)

یہ کمانڈ تمام اقسام کی فائلز یعنی ڈیٹا فائلز (دونوں رینڈم-Rendom اور سیقوئینشل-Sequential) اور پروگرام فائلز کو ختم کرنے کے لیے استعمال ہوتی ہے۔

مثالیں:

```
KILL "Inventory.bas"
```

یہ کمانڈ کرنٹ ڈائریکٹری میں Inventory.bas فائلز کو ختم کرتی ہے۔

```
KILL "G:\Goods\Inventory.*"
```

یہ کمانڈ کسی بھی ایکسٹینشن کے ساتھ Inventory نام کی تمام فائلز کو ختم کرتی ہے۔

KILL کمانڈ استعمال کرتے ہوئے محتاط رہیے۔ KILL کمانڈ کو استعمال کرتے ہوئے ہمیشہ فائل کے نام کی ایکسٹینشن کا ذکر کریں۔ آپ اس کمانڈ کے استعمال کے دوران حادثاتی طور پر ڈیٹا کھو سکتے ہیں۔

LIST Command 2.9.8

یہ کمانڈ سکرین، فائل یا پرنٹر پر پروگرام کو حصّوں میں یا مکمل طور پر دکھانے کے لیے استعمال ہوتی ہے۔

سینٹیکس Syntax

```
LIST [Line number] [-Line number] [,filename]
LIST [Line number-] [,File name]
```

وضاحت (Interpretation)

درج بالا سینٹیکس میں تمام پیرامیٹرز اختیاری ہیں جن کو اگر رہنے دیا جائے تو کمانڈ سب سے آخری درج شدہ پروگرام کی فہرست ظاہر کرتی ہے۔ [Line number] رینج صفر تا 65529 میں ایک درست لائن نمبر ہے۔ اگر LIST کمانڈ کے ساتھ فائل کا نام نہیں دیا جاتا تو آخری درج شدہ پروگرام کی مخصوص لائنوں کو لسٹ کیا جاتا ہے۔

مثالیں:

```
LIST
```

یہ کمانڈ پروگرام کی تمام لائنوں کی فہرست دکھاتی ہے۔

```
LIST-20
```

یہ کمانڈ پروگرام کی لائن نمبر 20 تک فہرست دکھاتی ہے۔

```
LIST 10-20
```

یہ کمانڈ لائن 10 تا لائن 20 تک کی فہرست دکھاتی ہے۔

```
LIST 20-
```

یہ کمانڈ لائن 20 سے پروگرام کے آخر تک فہرست دکھاتی ہے۔

LOAD Command 2.9.9

یہ کمانڈ فائل کو ڈسک سے میموری میں لوڈ کرتی ہے۔

Syntax سینٹیکس

LOAD filename [,r]

(Interpretation) وضاحت

filename اس فائل کا نام ہے جسے لوڈ کیا جاتا ہے۔ اگر [r] اوپشن کو LOAD کمانڈ کے ساتھ استعمال کیا جائے تو پروگرام لوڈ ہونے کے بعد چل پڑتا ہے اور تمام ڈیٹا فائلز کو کھلا رکھنا پڑتا ہے۔ ہم LOAD کمانڈ F3 کو دبا کر بھی ایگزیکیوٹ کر سکتے ہیں۔

مثال:

LOAD "D:\fact.bas"

یہ کمانڈ D ڈرائیو سے فائل جس کا نام fact.bas ہے کو لوڈ کرتی ہے۔

LOAD "D:\fact.bas", r

یہ کمانڈ D ڈرائیو سے فائل fact.bas کو لوڈ اور ایگزیکیوٹ کرتی ہے۔

MKDIR Command 2.9.10

یہ کمانڈ ایک سب ڈائریکٹری کو بنانے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

MKDIR pathname

(Interpretation) وضاحت

pathname اُس جگہ کی نشاندہی کرتا ہے جہاں سب ڈائریکٹری بنائی جاتی ہے۔ یہ ایک سٹرنگ ایکسپریشن (Expression) ہوتا ہے جسے 63 کریکٹرز سے زیادہ نہیں بڑھنا چاہیے۔

مثال:

MKDIR "D: Goods\Inventory"

یہ کمانڈ Goods نام کی ڈائریکٹری میں ایک سب ڈائریکٹری انوینٹری (Inventory) بناتی ہے۔

NAME Command 2.9.11

یہ کمانڈ فائل کا دوبارہ نام رکھنے کے لیے استعمال ہوتی ہیں۔

Syntax سینٹیکس

NAME Old filename As new-filename

(Interpretation) وضاحت

فائل کا نام تبدیل کر دیا جائے گا۔ پرانی فائل کے نام کو نئی فائل کے نام سے بدل دیا جاتا ہے۔

مثال:

NAME "Remarks .doc" AS "RMKS.doc"

یہ کمانڈ فائل Remarks.doc کو RMKS.doc کا نام دیتی ہے۔

RENUM Command 2.9.12

یہ کمانڈ پروگرام لائنز کو دوبارہ نمبر دینے کے لیے استعمال ہوتی ہے۔

سینٹیکس

Syntax

RENUM [new number], [old number] [,Increment]

وضاحت (Interpretation)

نیا نمبر نئی ترتیب کے لحاظ سے ابتدائی لائن نمبر ہوتا ہے۔ ڈیفالٹ (Default) نمبر 10 ہے۔ پرانا نمبر سے مراد موجودہ پروگرام میں وہ لائن ہے جہاں سے ری نمبرنگ شروع ہوتی ہے۔ ڈیفالٹ پروگرام کی پہلی لائن ڈیفالٹ نمبر ہوتی ہے۔ اضافہ سے مراد نئی ترتیب میں لائن نمبر کا اضافہ ہے۔ اضافہ کی موجودہ قیمت 10 ہے۔

مثالیں:

RENUM

یہ کمانڈ لائن نمبر 10 سے شروع ہو کر ہر ایک لائن میں 10 کے اضافہ سے نئے نمبر دیتی ہے۔

RENUM 80,30

یہ کمانڈ پورے پروگرام کو جو کہ لائن نمبر 80 سے شروع ہو رہا ہو لائن نمبرز میں 30 کے اضافہ کے ساتھ نئے نمبرز لگاتی ہے۔

RENUM 150, 70, 50

یہ کمانڈ لائن نمبر 70 سے پروگرام کے آخر تک لائنز کو نئے نمبر لگاتی ہے۔ نیا نمبر 150 سے شروع ہوگا (یعنی لائن نمبر 70 کا نمبر 150 کر دیا جائے گا) اور ہر اگلی لائن کے نمبر میں 50 کا اضافہ کیا جائے گا۔

RMDIR Command 2.9.13

یہ کمانڈ ڈسک سے ڈائریکٹری کو ہٹانے یا ختم کرنے کے لیے استعمال ہوتی ہے۔

سینٹیکس

Syntax

RMDIR pathname

وضاحت (Interpretation)

pathname موجود ڈائریکٹری کا پاتھ ہے جسے 63 کریکٹرز سے زیادہ نہیں بڑھنا چاہیے۔ جس ڈائریکٹری کو ختم کرنا ہو اُسے خالی ہونا چاہیے ورنہ ایرر کا پیغام ظاہر ہوگا۔

مثال:

RMDIR "D:\GOODS\INVENTORY"

ڈائریکٹری GOODS سے INVENTORY نام کی سب ڈائریکٹری کو ختم کرتی ہے۔

RUN Command 2.9.14

یہ کمانڈ میموری میں موجود پروگرام کو چلانے کے لیے استعمال ہوتی ہے۔ اگر پروگرام میموری میں موجود نہ ہو تو اُسے پہلے لوڈ کیا جاتا ہے اور پھر چلایا جاتا ہے۔

Syntax سینٹیکس

* RUN [line number] [, r]
* RUN filename [, r]

(Interpretation) وضاحت

موجودہ صورت میں RUN کمانڈ پروگرام کو ابتداء سے ایگزیکیوٹ کرنا شروع کر دیتی ہے۔ تاہم اگر لائن نمبر دیا گیا ہو تو پروگرام اُس مخصوص لائن نمبر سے ایگزیکیوٹ ہونا شروع ہو جاتا ہے۔ جب RUN کمانڈ کے ساتھ فائل نمبر دیا جاتا ہے تو یہ کمانڈ تمام کھلی ہوئی فائلوں کو بند کر دیتی ہے اور ڈسک سے مخصوص فائل کو لوڈ کرنے اور ایگزیکیوٹ کرنے سے پہلے میموری میں موجود تمام فہرست کو ختم کر دیتی ہے۔ "r" آپشن RUN کمانڈ کی ایگزیکیوشن کے دوران تمام فائلز کو کھلا رکھنے کے لیے استعمال کی جاتی ہے۔

مثال:

```
RUN "table.bas", r
```

یہ کمانڈ ڈیٹا فائلز کو بند کیے بغیر table.bas کو ایگزیکیوٹ کرتی ہے۔

اگر آپ کمپیوٹر پر سپیکر استعمال کر رہے ہیں تو نوٹ کیجیے کہ RUN کمانڈ کسی بھی آواز کو جو کہ اس وقت چل رہی ہو ایگزیکیوٹ ہوتے وقت ٹرن آف کر دیتی ہے اور موسیقی کو دوبارہ سیٹ کرتی ہے۔

SAVE Command 2.9.15

یہ کمانڈ ڈسک پر پروگرام کو محفوظ کرتی ہے تاکہ اسے بعد میں استعمال کیا جا سکے۔

Syntax سینٹیکس

```
SAVE filename [, a]
SAVE filename [, p]
```

(Interpretation) وضاحت

GW-BASIC ڈیفالٹ سیٹنگ (Setting) کے مطابق کمپریس (Compress) شدہ ثنائی فارمیٹ میں فائل کو محفوظ کرتی ہے۔ اگر آپشن [a] دیا گیا ہو تو فائل ASCII فارمیٹ میں محفوظ ہوتی ہے۔ آپشن [p] سے فائل این کوڈڈ (Encoded) ثنائی فارمیٹ میں محفوظ ہوتی ہے۔ آپ پروٹیکٹیڈ (Protected) فارمیٹ میں محفوظ فائل کو لسٹ یا ایڈٹ نہیں کر سکتے تاہم اس کو ایگزیکیوٹ کیا جا سکتا ہے۔

مثالیں:

```
SAVE "matrix.bas", a
```

ASCII فارمیٹ میں matrix.bas فائل کو محفوظ کرتی ہے۔

```
SAVE "matrix.bas", p
```

یہ کمانڈ این کوڈڈ ثنائی فارمیٹ میں فائل کو محفوظ کرتی ہے۔

SYSTEM Command 2.9.16

یہ کمانڈ جی ڈبلیو۔بیسک سے ونڈوز میں واپس آنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

SYSTEM

مثال: ڈائریکٹ موڈ میں ٹائپ کریں۔

SYSTEM

LLIST Command 2.9.17

Syntax سینٹیکس

LLIST [line number] [-line number]

LLIST [line number-]

(Interpretation) وضاحت

LLIST کمانڈ پر عمل کے بعد جی ڈبلیو۔بیسک کمانڈ لیول پر لوٹ آتا ہے۔LLIST کی کمانڈ سے پرنٹ ہونے والی لائنوں کی رینج کے اوپشنز لسٹ کمانڈ کے اوپشنز جیسے ہیں۔

LPRINT Command 2.9.18

یہ کمانڈ پرنٹر پر ڈیٹا پرنٹ کرنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

LPRINT [list of expressions] [;]

(Interpretation) وضاحت

ایکسپریشنز کی لسٹ سٹرنگ یا نومیرک ایکسپریشنز پر مشتمل ہوتی ہے جو کہ سیمی کولون کے ذریعے الگ کیے گئے ہوتے ہیں۔ سٹرنگ ایکسپریشنز سے مراد ایک سٹرنگ لٹرل یا ویری ایبل ہے جو خاص فارمیٹنگ (Formating) کریکٹرز پر مشتمل ہوتا ہے۔ فارمیٹنگ کریکٹرز پرنٹ شدہ سٹرنگز یا نمبرز کا فیلڈ اور فارمیٹ متعین کرتے ہیں۔

یہ سٹیٹمنٹ PRINT سٹیٹمنٹ جیسی ہی ہے سوائے اس کے کہ اس کمانڈ سے آؤٹ پٹ پرنٹر کو جاتا ہے۔سٹرنگ اور نومیرک فیلڈز کے بارے میں اور ان میں استعمال ہونے والے ویری ایبلز کے بارے میں زیادہ معلومات حاصل کرنے کے لیے PRINT سٹیٹمنٹ دیکھیں۔ایل پرنٹ سٹیٹمنٹ فرض کرتی ہے کہ پرنٹر چوڑائی میں 80 کریکٹرز تک پرنٹ کرسکتا ہے۔صفحے پر پرنٹ ہونے والے کریکٹرز کی تعداد نئے سرے سے سیٹ کرنے کے لیے (یہ فرض کرتے ہوئے کہ پرنٹر چوڑائی میں 80 کریکٹرز سے زیادہ پرنٹ کرسکتا ہے)، WIDTH کی سٹیٹمنٹ دیکھیں۔

CONT Command 2.9.19

یہ کمانڈ ایک وقفے کے بعد پروگرام کی ایگزیکیوشن جاری رکھنے کے لیے استعمال کی جاتی ہے۔

Syntax سینٹیکس

CONT

وضاحت (Interpretation)

CTRL-BREAK یا STOP کی کمانڈ سے پروگرام پر عمل روکنے کے بعد اُسے دوبارہ چلانے کے لیے یہ کمانڈ استعمال کی جاتی ہے۔ پروگرام پر عمل اُسی جگہ سے شروع ہوتا ہے جہاں سے رُکا تھا۔ اگر وقفہ INPUT سٹیٹمنٹ کے درمیان ہوا ہو تو پرومپٹ ظاہر ہوتا ہے اور اس کے بعد باقی پروگرام ایگزیکیوٹ کیا جاتا ہے۔

CONT کی کمانڈ ایررز درست کرنے کے لیے مفید ہے؛ اس صورت میں یہ ہمیں STOP کی سٹیٹمنٹ سے رُکنے، ڈائریکٹ سٹیٹمنٹس سے ویری ایبلز تبدیل کرنے، پروگرام پر عمل جاری رکھنے، یا کسی خاص لائن نمبر سے دوبارہ پروگرام چلانے کے لیے GOTO کے استعمال کا موقع فراہم کرتی ہے۔ اگر پروگرام لائن تبدیل ہو جاتی ہے تو CONT کی کمانڈ غیر مؤثر ہو جائے گی۔

2.10 بیسک سٹیٹمنٹس (BASIC Statements)

بیسک لینگوئج میں عام طور پر استعمال ہونے والی سٹیٹمنٹس درج ذیل ہیں۔

END Statement 2.10.1

یہ سٹیٹمنٹ پروگرام پر عمل روکنے کے لیے، تمام فائلیں بند کرنے کے لیے اور کمانڈ لیول پر واپس جانے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

END

REM Statements 2.10.2

یہ ایک ناقابلِ عمل سٹیٹمنٹ ہے اور یہ پروگرام میں توضیحی ریمارکس شامل کرنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

REM [remarks]

'[remarks]

مثال:

```
10      REM  This program calculates the average of two numbers
20      a = 15
30      b = 25
40      avg = (a + b) / 2
50      ' Display the average
60      PRINT "Average = "; avg
70      END
```

اس پروگرام میں لائن نمبر 10 اور لائن نمبر 50 ناقابلِ عمل ہیں۔ بیسک انٹرپریٹر ان پر کوئی عمل سرانجام نہیں دیتا؛ ٹرانسلیشن (Translation) کے عمل کے دوران ان کا مشین لینگوئج میں ترجمہ نہیں ہوتا۔ تاہم، REM Statement سے کوڈ پڑھنے میں آسانی

ہوجاتی ہے اور یہ پروگرام تبدیل کرنے، سمجھنے اور ایررز درست کرنے میں مدد دیتی ہے۔

STOP Statement 2.10.3

یہ سٹیٹمنٹ عارضی طور پر پروگرام کی ایگزیکیوشن روک دیتی ہے اور اُسے کمانڈ لیول پر لے آتی ہے۔

Syntax **سینٹیکس**

STOP

وضاحت (Interpretation)

STOP سٹیٹمنٹس پروگرام کی ایگزیکیوشن روکنے کے لیے پروگرام میں کہیں بھی استعمال ہوسکتی ہیں۔ جب پروگرام میں STOP سٹیٹمنٹ آتی ہے تو درج ذیل پیغام پرنٹ ہوتا ہے:

Break in line nnnnn

END سٹیٹمنٹ کے برعکس STOP سٹیٹمنٹ فائلیں بند نہیں کرتی۔ STOP سٹیٹمنٹ پر عمل کے بعد جی ڈبلیو- بیسک پروگرام ہمیشہ کمانڈ لیول پر آجاتا ہے۔ CONT کی کمانڈ دے کر پروگرام پر عمل دوبارہ شروع کیا جاتا ہے۔

```
Ok
auto
10 INPUT A, B, C
20 K = A 2* 5.3: L = B 3/0.26
30 STOP
40 M = C*K+100: PRINT M
50 END
Ok

RUN
? 1, 2, 3
Break in 30
Ok

PRINT L
 30.76923
Ok

Continue
 115.9
Ok
```

شکل 2.9: CONT کمانڈ کی مثال

2.11 بیسک میں اوپریٹرز (Operators in BASIC)

اوپریٹرز علامات ہوتی ہیں جو کہ ڈیٹا پر مختلف عوامل انجام دینے کے لیے استعمال کی جاتی ہیں۔ اِس میں اَرتھ میٹک، ریلیشنل (Relational)، لوجیکل (Logical) اور اسائنمنٹ اوپریٹرز شامل ہیں۔

2.11.1 اَرتھ میٹک اوپریٹرز (Arithmatic Operators)

اَرتھ میٹک اوپریٹرز، اعداد پر اَرتھ میٹک عوامل انجام دینے کے لیے استعمال کیے جاتے ہیں۔ GW-BASIC میں مندرجہ ذیل معیاری اَرتھ میٹک اوپریٹرز شامل ہیں۔

اوپریشنز	علامت	الجبری ایکسپریشن	بیسک ایکسپریشن
جمع	+	a+b	a+b
تفریق	-	a-b	a-b
ضرب	*	a×b	a*b
تقسیم	/	a/b	a/b
قُوّت نُما	^	a^n	a ^ n
نفی کرنا	-	-a	-a
ماڈولس(Modulus)	MOD	aMODb	aMODb
صحیح عددی تقسیم	\	a\b	a\b

پہلے چھ اوپریٹرز کا اِستعمال سیدھا ہے۔ آخری دو اوپریٹرز، ماڈولس (جو کہ ریمینڈر (Remainder) اوپریٹرز بھی کہلاتے ہیں) اور صحیح عددی تقسیم ہوتے ہیں۔ ڈویژن (Division) اوپریٹر کے برعکس، جو کہ حاصلِ تقسیم کو واپس کرتا ہے، ماڈولس اوپریٹر، عددی تقسیم میں باقی (remainder) بچنے والی رقم کو ظاہر کرتا ہے۔ مثال کے طور پر اگر a اور b دو اعداد ہوں جن کی قیمتیں بالترتیب 8 اور 3 ہوں تب aMODb کی قیمت 2 ہوگی جو کہ عددی تقسیم کا باقی (remainder) ہے۔ عددی تقسیم، حاصل تقسیم میں کسری قیمت کی اجازت نہیں دیتی۔ یہ ہمیشہ حاصلِ تقسیم کو مکمّل عدد کی شکل میں ظاہر کرتی ہے۔ مثال کے طور پر ایکسپریشن a\b (اُوپر کے دو متغیرات a اور b کے لیے) کا نتیجہ 2 ہوگا۔

2.11.2 ریلیشنل اوپریٹرز (Relational Operators)

ریلیشنل اوپریٹرز، دو قیمتوں کا موازنہ کرنے کے لیے اِستعمال کیے جاتے ہیں۔ یہ اوپریٹرز، ہمیشہ درست (true) یا غلط (false) کی شکل میں جواب دیتے ہیں۔ اگر ریلیشنل ایکسپریشن کا جواب درست ہو تو یہ ہمیشہ غیر صفری قیمت میں (زیادہ تر 1) جواب دیتے ہیں۔ اگر ریلیشنل ایکسپریشن کا جواب غلط ہو تو یہ ہمیشہ صفر قیمت دیتے ہیں۔

بیسک میں چھ بنیادی ریلیشنل اوپریٹرز ہوتے ہیں۔ فرض کریں کہ a,b اور c تین عددی متغیرات ہیں جن کی قیمتیں بالترتیب 123، 215 اور 123 ہیں۔

اوپریشن	علامت	ایکسپریشن (اظہار)	قیمت لگانا
برابر (موازنہ)	=	a = c	دُرست (غیر صفر)
سے کم	<	b < a	غلط (صفر)
سے زیادہ	>	a > c	غلط (صفر)
سے کم یا برابر	< =	a <= b	دُرست (غیر صفر)
سے زیادہ یا برابر	> =	b <= a	غلط (صفر)
کے برابر نہیں ہے	<>	a <> b	دُرست (غیر صفر)

2.11.3 منطقی اوپریٹرز (Logical Operators)

منطقی اوپریٹرز سادہ کنڈیشنز (Conditions) کو مزید پیچیدہ بنانے میں مدد دیتے ہیں (کنڈیشن سے مُراد، ایک ایکسپریشن ہے جو درست یا غلط کی نشاندہی کرے)۔ بیسک میں تین بنیادی لوجیکل اوپریٹرز ہوتے ہیں۔ یہ AND، OR اور NOT ہیں۔

```
Ok
auto
10 A$ = "Punjab Text"
20 B$ = "Book Board"
30 PRINT A$ + B$
40 END
50
Ok
RUN
Punjab TextBook Board
Ok
```

شکل 2.10: سٹرنگ ملانے کا عمل

پہلا منطقی اوپریٹر جو کہ "AND" ہے، جب دو کنڈیشنز کو ملاتا ہے تو اگر دونوں کنڈیشنز دُرست ہوں تو جواب دُرست ہوتا ہے ورنہ غلط ہونے کی نشاندہی کرتا ہے۔ دوسرا منطقی اوپریٹر "OR" جب دو کنڈیشنز کو ملاتا ہے تو اگر کنڈیشنز میں سے کوئی ایک دُرست ہو تو دُرست ہونے کی ورنہ غلط کی نشاندہی کرتا ہے۔ اسی طرح تیسرا منطقی اوپریٹر "NOT" جب ایک کنڈیشن پر لاگو کیا جائے تو یہ نشاندہی کے نتیجے کو اُلٹ دیتا ہے۔ اس کا مطلب ہے کہ اگر کنڈیشن دُرست نشاندہی کرے تو منطقی NOT اوپریٹر غلط ہونے کی نشاندہی کرتا ہے اور اسی طرح باقی بھی۔ ایک ایکسپریشن میں، منطقی عوامل، اَرتھ میٹک اور ریلیشنل عوامل کے بعد سرانجام دیئے جاتے ہیں۔ ان اوپریٹرز کی کارکردگی کو سمجھنے کے لیے مندرجہ ذیل جدول پر غور کریں۔

منطقی عمل	قیمت	قیمت	نتیجہ
NOT	X	-	NOT X
	T	-	F
	F	-	T
	X	Y	X AND Y
AND	T	T	T
	T	F	F
	F	T	F
	F	F	F
OR	X	Y	X OR Y
	T	T	T
	T	F	T
	F	T	T
	F	F	F

جدول 2.1: منطقی آپریٹرز کے نتائج

2.11.4 سٹرنگ ملانے والے اوپریٹرز (Concatenation Operators)

تمام ریلیشنل اوپریٹرز، موازنہ کرنے کے لیے سٹرنگز کے ساتھ استعمال کیے جاسکتے ہیں۔ علاوہ ازیں ریلیشنل اوپریٹرز کے علاوہ ایک اور اوپریشن، سٹرنگ ملانے کا آپریشن بھی سٹرنگ کانسٹینٹس اور متغیرات پر لاگو کیا جاسکتا ہے۔ سٹرنگ ملانے کے آپریشن کے لیے جمع کی علامت '+' استعمال ہوتی ہے اور یہ دو سٹرنگز کو ملاتی ہے۔ مندرجہ ذیل شکل کو دیکھئے جو کہ سٹرنگ ملانے کے عمل کو ظاہر کر رہی ہے۔

2.11.5 اسائنمنٹ اوپریٹر (Assignment Operator)

اسائنمنٹ اوپریٹر متغیر میں قیمت، سٹرنگ یا کمپیوٹیشنل نتیجہ کو ذخیرہ کرنے کے لیے استعمال ہوتا ہے۔ بیسک میں مساوی کی علامت (=) اسائنمنٹ اوپریٹر کو ظاہر کرتی ہے۔ جیسا کہ

```
variable_name = expression
```

جبکہ ایکسپریشن نومیرک یا سٹرنگ ایکسپریشن ہوسکتا ہے۔ جیسا کہ

```
a = 10
a$ = "Hello"
```

اوپریٹر کے دائیں طرف کی قیمت، اسائنمنٹ اوپریٹر کے بائیں طرف کے متغیر کو دی جاتی ہے۔ یہ سٹیٹمنٹ اسائنمنٹ سٹیٹمنٹ بھی کہلاتی ہے۔ نوٹ کریں کہ علامت '=' موازنہ کے لیے بھی استعمال ہوتی ہے لیکن اس کا دارومدار متن (سیاق وسباق) پر ہے کہ یہ کہاں استعمال کی جاتی ہے۔

2.11.6 اوپریٹر کی ترجیح (Operator Precedence)

آپریٹر کی ترجیح سے ایکسپریشن میں ان کے استعمال کی ترتیب کا علم ہوتا ہے۔ جدول 2.2 بیسک کے کچھ آپریٹرز کی بلند ترین سے کم ترین ترجیح کی فہرست پر مشتمل ہے۔

اوپریٹرز	ترجیح
()	بلند ترین
^	
_ (نفی کرنا)	
*,/	
\	
MOD	
+,-	
=,<>,<,<=,>,>=	
NOT	
AND	
OR	
= (اسائنمنٹ اوپریٹر)	کم ترین

جدول 2.2: اوپریٹر کی ترجیح

2.12 ٹائپ کنورژن (Type Conversion)

جب آپ کا پروگرام ایک قسم کی عددی قیمت کو دوسری تقسیم کے متغیر میں سٹور کرنے کی کوشش کرتا ہے، تو GW-BASIC مندرجہ ذیل اُصولوں کے مطابق ٹائپ کی تبدیلی (ٹائپ کنورژن) کا کام سرانجام دیتا ہے۔ اگر ایک قسم کے نومیرک کانسٹنٹ کو، مختلف قسم کے نومیرک ویری ایبل میں تبدیل کیا جاتا ہے تو عدد ویری ایبل کی ٹائپ کے مطابق تبدیل ہوتا ہے۔

مثال کے طور پر

```
10      LET x% = 51.39          'integer variable - storing a floating point value
20      PRINT x%
RUN
51
```

ایک ایکسپریشن کی قیمت نکالنے کے دوران ایک اَرتھ میٹک اور ریلیشنل اوپریشن میں تمام آپریینڈز کو کم ترین آپریینڈ کی ڈیٹا سٹوریج فارمیٹ (precision degree) میں منتقل کر دیا جاتا ہے۔ مثال کے طور پر

```
10      A # = 12 # 13
20      PRINT A #
RUN
9230769230769231
```

اَرتھ میٹک کو دو لفظی فارمیٹ میں سرانجام دیا جاتا ہے اور نتیجہ دو لفظی فارمیٹ کی شکل میں ہی #A میں واپس کیا جاتا ہے۔

```
10      A = 12 # 13
20      PRINT A
RUN
9230769
```

اَرتھ میٹک کو دو لفظی فارمیٹ میں سَرانجام دیا جاتا ہے اور نتیجہ کو ایک لفظی فارمیٹ کی شکل میں مختصر کیا جاتا ہے اور پھر A ویری ایبل کے آؤٹ پٹ کی شکل میں پرنٹ کیا جاتا ہے۔

جب ایک فلوٹنگ پوائنٹ قیمت ایک انٹیجر میں تبدیل کی جاتی ہے تو کسری حصہ راؤنڈ کر دیا جاتا ہے۔ مثال کے طور پر

```
10      NUM % = 23.67
20      PRINT NUM %
RUN
24
```

☆ ایک سٹرنگ متغیر کو نومیرک قیمت نہیں دی جا سکتی۔

2.13 اسائنمنٹ سٹیٹمنٹ (Assignment Statement)

GW-BASIC میں ایک ویری ایبل کو، ایک ایکسپریشن کی قیمت دینے کے دو طریقے ہیں۔ پہلے طریقہ میں، اسائنمنٹ اوپریٹر '=' کو

استعمال کرتے ہیں جیسا کہ ہم پچھلے متن میں پڑھ چکے ہیں اور دوسرے طریقہ میں LET سٹیٹمنٹ کو استعمال کرتے ہیں۔ حیرت انگیز طور پر یہ واحد ایسی سٹیٹمنٹ ہے جو لکھیں یا نہ لکھیں کام چل جاتا ہے لیکن اس کے پیرامیٹرز لازمی طور پر دینا پڑتے ہیں۔

Syntax **سینٹیکس**

[LET] variable = expression

وضاحت (Interpretation)

یہاں، لفظ LET اختیاری ہے لیکن ہمیں متغیر کا نام اور وہ ایکسپریشن جس کی قیمت ویری ایبل میں محفوظ کرنی ہے ضرور دینا چاہیے۔

مثال: ہم پہلے ہی LET سٹیٹمنٹ کی کچھ مثالیں دیکھ چکے ہیں۔ ایک دوسری مثال دیکھتے ہیں۔

```
10      REM Calculate Average of Two Numbers
20      LEM m = 24
30      LET n = 26
40      LET avg = (m+n) /2
50      PRINT "average ="; avg
60      END
RUN
average = 25
```

نوٹ: LET سٹیٹمنٹ کے استعمال کے بغیر بھی یہ پروگرام مکمل کیا جا سکتا ہے۔

ان پٹ/آؤٹ پٹ سٹیٹمنٹس (Input/Output Statements)

دوسری ہائی لیول پروگرامنگ لینگوئجز کی طرح، GW-BASIC ان پُٹ ڈیٹا کو سٹیٹمنٹس مہیا کرتا ہے اور نتائج دِکھاتا ہے۔ ان پُٹ، آؤٹ پُٹ سٹیٹمنٹس آپ کے پروگرام کو مزید قابلِ اَثر اور قابلِ استعمال بناتی ہیں۔ اس حصّے میں ہم GW-BASIC کی چند بنیادی ان پُٹ/آؤٹ پُٹ سٹیٹمنٹس پر تبصرہ کریں گے۔

2.14 ریڈ/ڈیٹا سٹیٹمنٹ (READ/DATA Statement)

یہ سٹیٹمنٹ نومیرک اور سٹرنگ کانسٹینٹس کو سٹور کرنے کے لیے استعمال کی جاتی ہے۔ اس مقصد کے لیے پروگرام میں کہیں READ سٹیٹمنٹ دی جاتی ہے۔

Syntax **سینٹیکس**

DATA coma-separated list of constants

وضاحت (Interpretation)

کانسٹنٹ سے مراد کوئی بھی سٹرنگ یا نومیرک کانسٹنٹ ہے۔ سٹرنگ کانسٹینٹس میں اگر کوما، کولن یا سپیسز (Spaces) ہوں تو انہیں ڈبل کوٹیشن مارکس میں لکھنا چاہیے۔ بصورتِ دیگر کوٹیشن مارکس کی ضرورت نہیں ہوتی۔ ڈیٹا سٹیٹمنٹ کے ساتھ کانسٹینٹس کی لسٹ ایک لائن سے نہیں بڑھ سکتی۔ اگر لسٹ اتنی لمبی ہے کہ ایک لائن میں نہیں آ سکتی تو باقی رہ جانے والے کانسٹنٹس کے لیے دوسری ڈیٹا سٹیٹمنٹ استعمال کرنی چاہیے۔ کئی ڈیٹا

سٹیٹمنٹس میں رکھے جانے والے ڈیٹا کے بارے میں تصور کیا جاتا ہے کہ یہ آئٹمز (Items) کی ایک لگاتار لسٹ ہے، اس بات سے قطع نظر کہ پروگرام میں DATA کی سٹیٹمنٹ کہاں ہے۔ READ سٹیٹمنٹ میں دیے گئے ویری ایبل کی ٹائپ ڈیٹا سٹیٹمنٹ میں متعلقہ کانسٹنٹ کے مطابق ہونی چاہیے۔

مثال:

```
10    This program demonstrates the use of DATA and READ statements
20    READ A, B, C$
30    PRINT C$; "=", (A+B)/2
40    DATA 10, 20, "Average"
50    END
RUN
Average = 15
```

READ سٹیٹمنٹ (READ Statement)

یہ سٹیٹمنٹ، ڈیٹا سٹیٹمنٹ سے قیمتوں کو پڑھتی ہے اور ان کو متعلقہ ویری ایبلز سے منسلک کرتی ہے۔

Syntax سینٹیکس

READ coma-separated list of variables

وضاحت (Interpretation)

READ سٹیٹمنٹ ڈیٹا سٹیٹمنٹ کا کمپلیمنٹری (Complementary) حصّہ ہے۔ READ اور DATA سٹیٹمنٹس ہمیشہ اکٹھی استعمال کی جاتی ہیں۔ READ سٹیٹمنٹ میں ویری ایبلز کی ایک فہرست ہوتی ہے اور یہ ان ویری ایبلز کی قیمتیں DATA سٹیٹمنٹ میں دیے گئے کانسٹینٹس سے پڑھتی ہے۔ اس کا مطلب ہے کہ DATA سٹیٹمنٹ میں کانسٹینٹس کی لسٹ سے READ سٹیٹمنٹ کے پہلے ویری ایبل کو پہلی قیمت دی جاتی ہے، دوسرے ویری ایبل کو دوسری قیمت دی جاتی ہے، تیسرے ویری ایبل کو تیسری قیمت دی جاتی ہے اور اسی طرح باقی بھی۔ تاہم ہمیں ڈیٹا سٹیٹمنٹ میں ویری ایبلز اور کانسٹینٹس کی اقسام سے متعلق بہت محتاط رہنا چاہیے۔ جیسا کہ نومیرک ویری ایبل کو نومیرک قیمت اور سٹرنگ ویری ایبل کو سٹرنگ قیمت دی جاتی ہے۔ اگر آپ کا پروگرام نومیرک ویری ایبل کو سٹرنگ قیمت دینے کی کوشش کرتا ہے یا سٹرنگ ویری ایبل کو نومیرک قیمت دیتا ہے تو "Type mismatch error" کا پیغام ظاہر ہوگا۔

اگر READ سٹیٹمنٹ میں ویری ایبلز تعداد میں DATA سٹیٹمنٹ کے کانسٹینٹس سے زیادہ ہوں تو READ سٹیٹمنٹ بہت سی DATA سٹیٹمنٹس سے قیمتوں کو پڑھ سکتی ہے۔ اسی طرح کئی READ سٹیٹمنٹس ایک DATA سٹیٹمنٹ سے قیمتوں کو پڑھ سکتی ہے۔ اگر ویری ایبلز کی فہرست میں ویری ایبلز کی تعداد DATA سٹیٹمنٹ میں ارکان کی تعداد سے زیادہ ہو تو Out of data کا پیغام ظاہر ہوتا ہے۔ اگر ویری ایبلز کی فہرست میں ویری ایبلز کی تعداد DATA سٹیٹمنٹ میں ارکان کی تعداد سے کم ہو تو اگلی READ سٹیٹمنٹ اس پہلے رکن سے ڈیٹا پڑھنا شروع کر دیتی ہے جسے ابھی تک پڑھا نہیں گیا۔ اگر اگلی ریڈ سٹیٹمنٹس موجود نہ ہوں تو زائد ڈیٹا کو چھوڑ دیا جاتا ہے۔

مثال:

```
10    REM  In this program one READ statement reads data from two DATA statements
20    REM This program calculates the perimeter of a pentagon, where a, bç,
30    REM d, and eare the lengths of sides of pentagon
40    READ A, B, C, D, E, P$
50    perimeter = a + b + c + d + e
60    PRINT  P$; " = "; perimeter
70    DATA  12, 18, 24
80    DATA  18, 35,perimeter
90    END
RUN
perimeter = 107
```

نوٹ: READ سٹیٹمنٹ کو پروگرام میں DATA سٹیٹمنٹ سے پہلے آنا چاہیے۔ اگر پروگرام کی فہرست میں DATA سٹیٹمنٹ پہلے آ رہی ہو تو READ سٹیٹمنٹ ڈیٹا کو پڑھ نہیں سکتی۔

2.15 ریسٹور سٹیٹمنٹ (RESTORE Statement)

یہ سٹیٹمنٹ READ سٹیٹمنٹ کے ذریعے DATA سٹیٹمنٹ کو دوبارہ پڑھنے کا باعث بنتی ہے (اگر اسے پہلے پڑھا جا چکا ہو)۔

Syntax سینٹیکس

```
RESTORE [line number]
```

وضاحت (Interpretation)

لائن نمبر سے مراد اس DATA سٹیٹمنٹ کا لائن نمبر ہے جس کو دوبارہ پڑھا جانا ہو۔ اگلی READ سٹیٹمنٹ متعلقہ DATA سٹیٹمنٹ میں پہلی آئیٹم کو پڑھتی ہے۔ ہم لائن نمبر کو چھوڑ سکتے ہیں۔ اگر ایسا ہے تو اگلی READ سٹیٹمنٹ پہلی DATA سٹیٹمنٹ میں پہلی آئیٹم کو پڑھتی ہے۔

مثال:

```
10    READ A,B,C
20    RESTORE
30    READ x, y, z
40    PRINT A, B, C
50    PRINT
60    PRINT x, y, z
70    DATA 10, 20, 30
80    DATA 40, 50, 60
```

```
90      END
RUN
10              20              30
10              20              30
```

نوٹ: مندرجہ بالا مثال سے صاف ظاہر ہے کہ دوسری READ سٹیٹمنٹ دوسری DATA سٹیٹمنٹ کو نہیں پڑھتی۔ اس کی بجائے RESTORE پہلی DATA سٹیٹمنٹ کو اگلی READ سٹیٹمنٹ کے ذریعے دوبارہ پڑھے جانے کے لیے سیٹ کرتی ہے۔

2.16 اِن پُٹ سٹیٹمنٹ (INPUT Statement)

یہ سٹیٹمنٹ پروگرام کی ایگزیکیوشن کے دوران یوزر سے ڈیٹا اِن پُٹ کرنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

☆ INPUT [;] [prompt string;] comaseparated list of variables

☆ INPUT [;] [prompt string,] comaseparated list of variables

وضاحت (Interpretation)

پرومپٹ سٹرنگ ایک پیغام ہے جو کہ یوزر کی مدد کے لیے سکرین پر ظاہر ہوتا ہے تا کہ وہ درست ڈیٹا اِن پُٹ کر سکے۔ ہم ایک اِن پُٹ سٹیٹمنٹ کے ساتھ ایک سے زیادہ ویری ایبلز دے سکتے ہیں۔ پروگرام کی ایگزیکیوشن کے دوران یوزر جو قیمتیں دیتا ہے وہ متناظرہ ویری ایبلز کو اُسی ترتیب میں دی جاتی ہیں جس میں انھیں فہرست کیا گیا ہو۔ فہرست میں موجود ویری ایبلز کی تعداد اور ڈیٹا آئیٹمز کی تعداد جنہیں یوزر مہیا کرتا ہے برابر ہوتی ہے۔ ہر ڈیٹا آئیٹم اِن پُٹ کی ٹائپ ویری ایبل کی ٹائپ جیسی ہونی چاہیے۔

جب ویری ایبلز کی فہرست سے پرومپٹ سٹرنگ کو علیٰحدہ کرنے کے لیے کولن (:) استعمال کیا جاتا ہے تو پرومپٹ سٹرنگ کے آخر پر ایک سوالیہ نشان (?) ظاہر ہوتا ہے۔ اس سوالیہ نشان سے بھی کولن کی بجائے قومہ استعمال کرتے ہوئے بچا جا سکتا ہے۔

مثال:۔ درج ذیل شکل INPUT سٹیٹمنٹ کے استعمال کو ظاہر کرنے والے ایک پروگرام پر مشتمل ہے۔

```
10 REM This program calculates the circumference of the circle
20 INPUT "Enter the radius of the circle >", RADIUS!
30 CIRCUMFERENCE# = 2 * 3.14 * RADIUS!
40 PRINT "Circumference = "; CIRCUMFERENCE#
50 END
Ok
RUN
Enter the radius of the circle >2.7
Circumference =  16.95600128173828
Ok
```

شکل نمبر 2.11: اِن پُٹ سٹیٹمنٹ کی مثال

جب پروگرام کی ایگزیکیوشن کے دوران اِن پُٹ سٹیٹمنٹ آئے تو پروگرام رُک جاتا ہے، پرومپٹ سٹرنگ ظاہر ہوتا ہے اور یوزر مطلوبہ ڈیٹا ٹائپ کرتا ہے۔ سٹرنگز جو کہ اِن پُٹ سٹیٹمنٹ کو اِن پُٹ کرتے ہیں کے گرد کوٹیشن مارکس کی ضرورت نہیں ہوتی جب تک کہ وہ قومے (commas) یا خالی جگہوں (blanks) پر مشتمل نہ ہوں۔

2.17 پرنٹ سٹیٹمنٹ (PRINT Statement)

یہ بیسک میں کثرت سے استعمال ہونے والی سٹیٹمنٹ ہے۔ بیسک میں تقریباً ہر پروگرام اس کو استعمال کرتا ہے۔ اس سبق میں اب تک ہم نے جو پروگرامز دیکھے ہیں، ان میں اس کو مختلف طریقوں سے استعمال کیا گیا ہے۔ یہ سٹیٹمنٹ متن اور نمبروں کو سکرین پر دکھانے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

```
PRINT [list of expressios] [;]

?[list of expressios] [;]
```

وضاحت (Interpretation)

فہرست میں ایکسپریشنز نومیرک اور/یا سٹرنگ ایکسپریشنز ہو سکتے ہیں جنہیں قومے، سپیسز (spaces) یا سیمی کولن لگا کر علیٰحدہ کیا جاتا ہے۔ اگر ہم ایکسپریشنز کی فہرست کو خارج کرتے ہیں تو ایک خالی لائن پرنٹ ہوتی ہے۔ ہم پرنٹ سٹیٹمنٹ کی جگہ ایک سوالیہ نشان (؟) استعمال کر سکتے ہیں۔ یہ اُسی طرح کام کرے گا جس طرح کہ پرنٹ سٹیٹمنٹ کرتی ہے۔

جب پرنٹ اسٹیٹمنٹ میں دو ایکسپریشنز کو ایک سیمی کولن (;) سے علیٰحدہ کیا جاتا ہے تو دوسرا ایکسپریشن سکرین پر پہلے ایکسپریشن کے بعد دکھایا جاتا ہے۔ GW-BASIC ہر لائن کو 14 جگہوں (spaces) کی پرنٹ زون (Print zone) میں تقسیم کرتی ہے۔ ہم ایکسپریشنز کو علیٰحدہ کرنے کے لیے سیمی کولن کی جگہ قومہ استعمال کر سکتے ہیں۔ اس سے دوسرا ایکسپریشن اگلے زون کے شروع میں ظاہر ہوتا ہے۔

مثال:۔ درج ذیل شکل پرنٹ سٹیٹمنٹ کے استعمال کو ظاہر کرتی ہے۔

```
Ok
LIST
10 A$= "SINDH": B$= "PUNJAB": C$= "BALOCHISTAN": D$= "NWFP": E$= "KASHMIR"
20 PRINT A$; B$; C$; D$; E$
30 PRINT
40 PRINT A$, B$, C$, D$, E$
50 END
Ok

RUN
SINDHPUNJABBALOCHISTANNWFPKASHMIR

SINDH          PUNJAB         BALOCHISTAN    NWFP           KASHMIR
Ok
```

```
Ok
LIST
10 A=10 : B=20 : C=30
20 CITY$ = "Lahore": COUNTRY$ = "Pakistan"
30 PRINT A, B, C
40 PRINT A, B, C, CITY$, COUNRY$
Ok

RUN
 10             20             30
 10             20             30             Lahore         Pakistan
Ok
```

شکل نمبر 2.11: پرنٹ سٹیٹمنٹ کی مثالیں

2.18 پرنٹ یوزنگ سٹیٹمنٹ (PRINT USING Statement)

یہ سٹیٹمنٹ ایک مخصوص فارمیٹ میں سکرین پر نمبروں اور سٹرنگز کو دکھانے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

PRINT USING String expression, list of expressions [;]

وضاحت (Interpretation)

سٹرنگ ایکسپریشنز سٹرنگ لیٹرل (Literal) یا ویری ایبل ہوتا ہے جو کہ مخصوص فارمیٹنگ کریکٹرز پر مشتمل ہوتا ہے۔ فارمیٹنگ کریکٹرز پرنٹڈ (Printed) سٹرنگز کے فیلڈ اور فارمیٹ یا نمبروں کو بیان کرتا ہے۔ ایکسپریشنز کی فہرست سٹرنگ یا نومیرک ایکسپریشنز پر مشتمل ہوتی ہے جنہیں سیمی کولن سے علیٰحدہ کیا گیا ہوتا ہے۔

سٹرنگ فیلڈز (String Fields)

سٹرنگ فیلڈز کو فارمیٹ کرنے کے درج ذیل تین کریکٹرز استعمال ہو سکتے ہیں۔

☆ ! اس کا مطلب ہے کہ سٹرنگ میں صرف پہلے کریکٹر کو پرنٹ کرنا ہے۔

☆ \n spaces\ اس کا مطلب ہے کہ سٹرنگ سے 2+n کریکٹرز کو پرنٹ کیا جانا ہے۔ اگر بیک سلیشز کو بغیر سپیسز کے ٹائپ کیا جاتا ہے تو دو کریکٹرز پرنٹ ہوتے ہیں؛ اگر بیک سلیشز ایک سپیس کے ساتھ ٹائپ کی جاتی ہیں تو تین کریکٹرز پرنٹ ہوتے ہیں اور اس طرح مزید بھی۔

```
10      AS= "Work": BS="Hard"
30      PRINT USING "!"; A$; B$
40      PRINT USING "\\"; A$; B$
50      PRINT USING "\\"; A$; B$; "!!"
RUN
WH
Work Hard
Work Hard !!
```

☆ & یہ کریکٹر ویری ایبل کی لمبائی کے سٹرنگ فیلڈ کو مخصوص کرتا ہے۔ جب فیلڈ کو & کے ساتھ مخصوص کیا جاتا ہے تو سٹرنگ آؤٹ پُٹ مکمل طور پر ان پُٹ کی طرح ہوتی ہے۔

نومیرک فیلڈز (Numeric Fields)

نومیرک فیلڈ فارمیٹ کرنے کے لیے درج ذیل خاص کریکٹرز استعمال ہو سکتے ہیں۔

ہر ہندسے کی پوزیشن کو ظاہر کرنے کے لیے پاؤنڈ کی علامت استعمال ہوتی ہے۔ ہندسے کی پوزیشنز ہمیشہ پُر کی جاتی ہیں۔ اگر پرنٹ کیے جانے والے عدد میں مخصوص پوزیشنز کے مقابلہ میں ہندسے کم ہوں تو نمبر فیلڈ میں دائیں طرف سیٹ (Right Justified) ہو جاتا ہے۔

نقطہ اعشاریہ فیلڈ میں کہیں بھی دیا جا سکتا ہے۔ اگر فارمیٹ سٹرنگ ذکر کرتا ہے کہ ہندسہ کو نقطہ اعشاریہ سے پہلے آنا ہے تو ہندسہ کو ہمیشہ

پرنٹ کیا جاتا ہے (صفر اگر ضروری ہو تو)۔ نمبروں کو اگر ضروری ہو تو راؤنڈ آف (Round off) کیا جاتا ہے۔ مثال کے طور پر

```
PRINT USING "##.##"; .85
0.85
PRINT USING "###.##";254.753
254.75
PRINT USING "##.##"; 10.2, 5.3, 66.789, .234
10.20, 5.30, 66.790, .23
```

آخری مثال میں فارمیٹ سٹرنگ کے آخر پر تین سپیسز کو لائن پر پرنٹڈ قیمتوں کو علیٰحدہ کرنے کے لیے داخل کیا گیا ہے۔

☆ فارمیٹ سٹرنگ کے آغاز یا آخر پر جمع کی علامت کی وجہ سے نمبر کے ساتھ اس سے پہلے یا بعد میں (مثبت یا منفی) علامت لگتی ہے۔

☆ فارمیٹ سٹرنگ میں نقطہ اعشاریہ کے بائیں طرف قومہ نقطہ اعشاریہ کے بائیں طرف ہر تیسرے ہندسے کے بعد قومہ پرنٹ کیے جانے کا سبب بنتا ہے۔ فارمیٹ سٹرنگ کے آخر پر قومہ سٹرنگ کے حصہ کے طور پر پرنٹ ہوتا ہے۔

مشق

1- خالی جگہیں پُر کریں۔

(i) بیسک سے مراد ہے۔۔۔۔۔

(ii) بیسک لینگوئج جان کیمنی اور تھامس کرٹز نے ڈارماؤتھ کالج، امریکہ میں۔۔۔۔۔ میں بنائی۔

(iii) جی ڈبلیو-بیسک جب لوڈ ہوتی ہے تو۔۔۔۔۔ کا پیغام ظاہر کرتی ہے۔

(iv) ۔۔۔۔۔ موڈ میں، کمانڈز ٹائپ ہونے کے ساتھ ہی ایگزیکیوٹ ہوتی ہیں۔

(v) بیسک پروگرام کی ہر سٹیٹمنٹ کے شروع میں ایک۔۔۔۔۔ آتا ہے۔

(vi) جی ڈبلیو-بیسک پروگرام کی ایک لائن میں۔۔۔۔۔ سے زیادہ کریکٹرز نہیں آ سکتے۔

(vii) جی ڈبلیو-بیسک پروگرام میں زیادہ سے زیادہ۔۔۔۔۔ قطاریں آ سکتی ہیں۔

(viii) جی ڈبلیو-بیسک پروگرام کی موجودہ ایکسٹینشن۔۔۔۔۔ ہے۔

(ix) ایسے الفاظ جن کا پروگرامنگ لینگوئج میں پہلے سے مفہوم دیا گیا ہوتا ہے۔۔۔۔۔ کہلاتے ہیں۔

(x) ۔۔۔۔۔ ناقابل عمل سٹیٹمنٹ کی ایک مثال ہے۔

2- درست جواب منتخب کریں۔

(i) جی ڈبلیو-بیسک آپریٹ کر سکتی ہے

(a) ایک موڈ میں (b) دو موڈز میں

(c) تین موڈز میں (d) کئی موڈز میں

(ii) جی ڈبلیو-بیسک میں ایک ویری ایبل کی زیادہ سے زیادہ لمبائی ہے

(a) 31 (b) 32

(c) 40 (d) 45

(iii) اگر ایک قطار میں دو یا دو سے زیادہ سٹیٹمنٹس لکھی جاتی ہیں تو انہیں ۔۔۔۔۔ سے جدا کرنا چاہیے

(a) کالن (b) سیمی کالن

(c) کوما (d) ہائفن

(iv) انٹیجر ویری ایبلز کی ٹائپ کی نشاندہی کے لیے کون سا کریکٹر استعمال ہوتا ہے؟

(a) ! (b) %

(c) # (d) $

(v) ان میں سے کون سا آپریٹر سب سے زیادہ ترجیحی قیمت رکھتا ہے؟

(a) ^ (b) *

(c) + (d) =

(vi) ویری ایبل کا نام شروع ہونا چاہیے

(a) ایلفابیٹ سے (b) انڈرسکور سے

(c) ڈیجٹ سے (d) ایلفابیٹ یا انڈرسکور سے

(vii) جی ڈبلیو-بیسک میں پروگرام چلانے کے لیے ان میں سے کون سی شارٹ کی استعمال ہوتی ہے

(a) F4 (b) F3

(c) F2 (d) F1

(viii) جب ایک فلوٹنگ پوائنٹ ویلیو کو انٹیجر میں تبدیل کیا جاتا ہے تو کسری حصہ کو

(a) ختم کر دیتے ہیں (b) قریبی بڑے ہندسے میں تبدیل کر دیتے ہیں

(c) ختم کر دیتے ہیں یا قریبی بڑے ہندسے میں تبدیل کر دیتے ہیں (d) تبدیلی ممکن نہیں ہے

(ix) درج ذیل میں سے کون سی سٹیٹمنٹ کسی پروگرام کی ایگزیکیوشن وقتی طور پر روک دیتی ہے؟

(a) BREAK (b) END

(c) PAUSE (d) STOP

(x) درج ذیل میں سے کون سی کمانڈ اس پروگرام کو چلاتی ہے جس کی ایگزیکیوشن عارضی طور پر روکی گئی تھی؟

(a) CONTINUE (b) CONT

(c) RESTART (d) START

3- صحیح/غلط کی نشاندہی کریں۔

(i) کوئیک بیسک مینیوز پر مشتمل ماحول مہیا کرتی ہے۔

(ii) بیسک کی ہر کمانڈ کے شروع میں لائن نمبر آتا ہے۔

(iii) KILL کی کمانڈ سسٹم میں جاری عمل کو ختم کرنے کے لیے استعمال ہوتی ہے۔

(iv) DELETE کی کمانڈ فائل ختم کرنے کے لیے استعمال ہوتی ہے۔

(v) AUTO کی کمانڈ اپنے آپ لائن نمبر لگانے کے لیے استعمال ہوتی ہے۔

(vi) ؟علامت PRINT کمانڈ کی جگہ پر استعمال ہوسکتی ہے۔

(vii) جی ڈبلیو بیسک میں F1 کی مدد حاصل کرنے کے لیے استعمال کی جاتی ہے۔

(viii) پروگرام کو ایگزیکیوشن سے پہلے لوڈ کرنا ضروری ہے۔

(ix) READ سٹیٹمنٹ کی بورڈ سے ڈیٹا لیتی ہے۔

(x) پروگرام کی ایگزیکیوشن کے دوران کسی کانسٹینٹ کی قیمت تبدیل نہیں کی جاسکتی۔

4- GW-BASIC کو کتنے موڈز میں چلایا جاسکتا ہے؟ مختصر طور پر بیان کریں۔

5- GW-BASIC میں ویری ایبلز کو نام دینے کے قوانین بیان کریں۔

6- ٹائپ ڈکلیریشن کریکٹرز کیا ہیں؟ مثالوں سے ان کے استعمال کی وضاحت کریں۔

7- مختصر طور پر حسابی، منطقی اور ریلیشنل اوپریٹرز کو بیان کیجیے۔

8- ٹائپ کنورژن سے کیا مُراد ہے؟ بیسک میں ٹائپ کنورژن کے قوانین بیان کریں۔

9- DATA STATEMENT میں دی گئی دس قیمتیں پڑھنے کے لیے ایک پروگرام لکھیں اور ان قیمتوں کی حاصل جمع سکرین پر ظاہر کریں۔

10- درج ذیل مختصر سوالات کے جواب دیں۔

(i) GW-BASIC میں F1 تا F9 تک کی فنکشن کیز کے کام بیان کریں۔

(ii) IDE سے کیا مراد ہے؟ GW-BASIC میں IDE کی خصوصیات بیان کریں۔

(iii) 'پروگرام لوڈ کرنا' کی اصطلاح کی وضاحت کریں۔ پروگرام چلانے سے پہلے کیوں لوڈ کرنا چاہیے؟

(iv) بیسک کمانڈز اور سٹیٹمنٹس میں فرق بیان کریں۔

(v) CLEAR کی کمانڈ اور CLS کی کمانڈ میں کیا فرق ہے؟

(vi) درج ذیل کمانڈز کا مقصد اور سینٹیکس بیان کریں۔

(a) DELETE (b) KILL (c) FILES (d) LIST

(e) LOAD (f) SYSTEM (g) NAME (h) RENUM

(i) RUN (j) SAVE

(vii) بیسک پروگرام کی بناوٹ مختصر طور پر بیان کریں۔

(viii) ویری ایبل اور کانسٹینٹ میں فرق بیان کریں۔

(ix) ایک پروگرام لکھیں جو کہ جماعت دہم کے طالب علم کا نام، رولنمبر، جماعت، سیکشن اور مختلف مضامین میں نمبر پوچھتا ہے۔ پروگرام طالب علم کے کل نمبر اور فیصد شمار کر کے بتائے گا۔ [اشارہ: یوزر سے ڈیٹا لینے کے لیے ان پٹ کی سٹیٹمنٹ استعمال کریں۔ فرض کریں کہ کل نمبر 850 ہیں۔]

(x) v میٹر فی سیکنڈ کی رفتار سے حرکت کرتی ہوئی کار کا t سیکنڈ میں طے کردہ فاصلہ معلوم کرنے کے لیے پروگرام لکھیں۔ پروگرام کو ان پٹ کے طور پر اوسط رفتار اور وقت معلوم کرنا چاہیے۔ [v اور t کی قیمتیں لینے کے لیے INPUT سٹیٹمنٹ استعمال کریں۔ آپ گزشتہ باب کی مشق میں اس پروگرام کے لیے الگورتھم لکھ چکے ہیں۔]

(xi) PRINT سٹیٹمنٹ کے ساتھ کوما (,) اور سیمی کالن (;) کے استعمال کی وضاحت کے لیے ایک مثال دیں۔

-11 ایک سلنڈر کا حجم معلوم کرنے کے لیے پروگرام لکھیں۔ پروگرام کو چاہیے کہ وہ یوزر سے INPUT سٹیٹمنٹ کے ذریعے سلنڈر کی بلندی اور اس کے پیندے کا رداس معلوم کرے۔ [اشارہ: بلندی × رداس × رداس × 3.14 = حجم]

-12 دیے گئے ہندسے کا مربع معلوم کرنے کے لیے پروگرام تحریر کریں۔ پروگرام کو چاہیے کہ یوزر سے INPUT سٹیٹمنٹ کے ذریعے ہندسہ معلوم کرے۔

-13 LET سٹیٹمنٹ کی مدد سے تین اعداد کی حاصل جمع اور اوسط نکالنے اور پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

جوابات

-1 (i) Beginners All Purpose Symbolic Instruction Code

(ii) 1963 (iii) Ok (iv) ڈائریکٹ (v) لائن نمبر (vi) 255

(vii) 65529 (viii) bas (ix) کی ورڈز/ریزرو ورڈز (x) REM

-2 (i) b (ii) c (iii) a (iv) b (v) a

(vi) a (vii) c (viii) b (ix) d (x) b

-3 (i) صحیح (ii) غلط (iii) غلط (iv) غلط (v) صحیح

(vi) صحیح (vii) غلط (viii) صحیح (ix) غلط (x) صحیح

باب 3

کنٹرول سٹرکچرز

(Control Structures)

کنٹرول سٹرکچرز پروگرام پر عمل کو کنٹرول کرتی ہیں۔ بیسک میں تین قسم کے کنٹرول سٹرکچرز استعمال ہوتے ہیں؛ یہ ترتیب (Sequence)، چناؤ (Selection) اور لُوپ (Loop) ہیں۔ تمام پروگرامز منطق کو لاگو کرنے کے لیے ان کنٹرول سٹرکچرز کو استعمال کرتے ہیں۔

اب تک ہم صرف ترتیب سٹرکچر کو استعمال کرتے رہے ہیں۔ ترتیب سٹرکچر میں ہدایات کو لائن نمبرز کے بڑھتے ہوئے آرڈر (Order) کے حساب سے ایگزیکیوٹ کیا جاتا ہے۔ لہٰذا چھوٹی لائن نمبرز والی ہدایات کو بڑی لائن نمبرز والی ہدایات سے پہلے ایگزیکیوٹ کیا جاتا ہے۔ مثال کے طور پر درج ذیل پروگرام ترتیب سٹرکچر کو ظاہر کرتا ہے۔

```
10    REM This program does not transfer control to any system.
20    REM conditionally or unconditionally. the Statementa are
30    REM executed in the same sequence in which they are written.
40    R =10.5
50    AREA = 3.14 * R * R
60    PRINT "Area = "; AREA
70    END
```

Output

346.185

GW-BASIC میں پروگرام کی ایگزیکیوشن کے دوران پروگرام کنٹرول کو مشروط یا غیر مشروط طور پر پروگرام کے ایک حصّہ سے دوسرے حصّہ میں منتقل کیا جاسکتا ہے۔ GW-BASIC دونوں اقسام کے کنٹرول ٹرانسفر کے لیے سٹیٹمنٹس مہیا کرتا ہے۔

☆ غیر مشروط کنٹرول کے ٹرانسفر میں پروگرام کنٹرول کسی شرط کے بغیر ایک مخصوص لائن یا ایک سے زیادہ لائنز کو چھوڑ کر کسی خاص لائن کو منتقل ہو جاتا ہے۔

☆ مشروط کنٹرول کے ٹرانسفر میں پروگرام کنٹرول کسی خاص شرط کے تحت ایک مخصوص لائن یا ایک سے زیادہ لائنز کو چھوڑ کر کسی خاص لائن کو منتقل ہو جاتا ہے۔

3.1 کنٹرول کا غیر مشروط ٹرانسفر (Unconditional Transfaer of Control)

کنٹرول کے غیر مشروط ٹرانسفر میں کنٹرول کسی شرط کے بغیر پروگرام کے ایک حصّہ سے دوسرے حصّہ کو منتقل ہو جاتا ہے۔ GW-BASIC میں، کنٹرول غیر مشروط طور پر منتقل کرنے کے لیے GO TO سٹیٹمنٹ استعمال ہوتی ہے۔

میں، کنٹرول غیر مشروط طور پر منتقل کرنے کے لیے GO TO سٹیٹمنٹ استعمال ہوتی ہے۔

GOTO Statement 3.1.1

GOTO سٹیٹمنٹ پروگرام کی کسی ایک لائن سے کسی خاص لائن کو بغیر کسی شرط کے کنٹرول ٹرانسفر کرنے کے لیے استعمال ہوتی ہے۔

Syntax سینٹیکس

GOTO *line number*

وضاحت (Interpretation)

لائن نمبر پروگرام میں ایک درست لائن نمبر ہوتا ہے۔ پروگرام کنٹرول بغیر کسی شرط کو ٹیسٹ کیے مخصوص لائن نمبر کو منتقل ہو جاتا ہے۔ یہ نارمل (Normal) پروگرام فلو میں رکاوٹ کا باعث بنتا ہے۔ اسے پروگرامنگ کے جدید انداز کے مطابق اچھی مشق تصور نہیں کیا جاتا۔ یہی وجہ ہے کہ اکثر نئی پروگرامنگ لینگویجز میں GOTO سٹیٹمنٹ کے استعمال کی حوصلہ شکنی کی جاتی ہے۔ اگر GOTO سٹیٹمنٹ ایک مخصوص لائن نمبر پر قابلِ عمل (Executable) سٹیٹمنٹ ہو تب اس سٹیٹمنٹ اور اس کے بعد آنے والی سٹیٹمنٹس پر عمل کیا جاتا ہے۔ بصورتِ دیگر اس سٹیٹمنٹ کے بعد آنے والی پہلی قابلِ عمل سٹیٹمنٹ پر عمل شروع ہو جاتا ہے۔

مثال: درج ذیل مثالیں GOTO سٹیٹمنٹ کے استعمال کو ظاہر کرتی ہیں۔

```
10    READ A,B,X,Y
20    GOTO 60
30    LET X = X*X+A
40    LET Y = Y*Y+B
50    PRINT X,Y
60    REM Because of unconditional transfer of control, X and Y will
65    REM not be calculated in the above lines.
70    LET X = A*X
80    LET Y  = B*Y
90    PRINT X,Y
100   DATA 6,3,4,5
110   END
```

یہ مثال ظاہر کرتی ہے کہ لائن نمبر 10 سے A,B,X,Y کی قیمتیں پڑھنے کے بعد کنٹرول لائن نمبر 60 کو منتقل ہو جاتا ہے جہاں اس کا سامنا REM سے ہوتا ہے (جو کہ ایک ناقابلِ عمل سٹیٹمنٹ ہے)۔ پھر کنٹرول لائن نمبر 70 کو منتقل ہو جاتا ہے جہاں X اور Y کی قیمتیں شمار کی جاتی ہیں۔ پھر لائن نمبر 90 پر X اور Y کی قیمتیں پرنٹ کی جاتی ہیں اور پروگرام ختم ہو جاتا ہے۔ سٹیٹمنٹس نمبر 30, 40 اور 50 پر عمل کیے بغیر ان سے آگے نکل جاتے ہیں۔ اگر ہم لائن نمبرز 40,30 اور 50 کو ایگزیکیوٹ کرنا چاہتے ہیں تو ہم چند مزید

جمپ سٹیٹمنٹس یعنی GOTO سٹیٹمنٹس کی ضرورت ہوگی۔ درج ذیل مثال پر غور کریں۔

```
10    READ A, B, X, Y
20    GOTO 60
30    LET X = X*X+A
40    LET Y = Y*Y+B
50    PRINT X,Y
55    GOTO 100
60    REM Because of unconditional transfer of control, X and Y will
65    REM not be calculated in the above lines.
70    LET X = A*X
80    LET Y = B*Y
90    PRINT X,Y
95    GOTO 30
100   DATA 6,3,4,5
110   END
```

اس طرح لائن نمبر 55 اور 95 دینے سے تمام سٹیٹمنٹس پر عمل ہو جاتا ہے لیکن اس سے پروگرام قدرے پیچیدہ ہو جاتا ہے۔ لہٰذا لائن نمبر 30 اور 40 سے X=X*X+A اور Y = Y*Y+B کے ایکسپریشنز کے استعمال سے X اور Y کی قیمتیں معلوم کی جاتی ہیں اور لائن نمبر 70 اور 80 سے X = A*X اور Y = B*Y کے ایکسپریشنز کے استعمال سے X اور Y کی قیمتیں معلوم کی جاتی ہیں۔

3.2 کنٹرول کا مشروط ٹرانسفر (Conditional Transfer of Control)

کنٹرول کے مشروط ٹرانسفر سے مراد کسی خاص شرط کی بنا پر پروگرام کے ایک حصے سے دوسرے حصے کو کنٹرول کا منتقل کرنا ہے۔ GW-BASIC میں پروگرام کے ایک حصہ سے دوسرے حصہ تک مشروط طور پر کنٹرول ٹرانسفر کرنے کے لیے بہت سی سٹیٹمنٹس ہیں۔ یہاں ہم ان پر مختصراً بحث کریں گے۔

3.2.1 ON... GOTO Statements

یہ کثیر الشاخی (multiple branching) سٹیٹمنٹ ہے۔ GOTO سٹیٹمنٹ کے برعکس جو کہ صرف ایک ٹرانسفر پوائنٹ (Transfer point) کی اجازت دیتی ہے۔ ON ... GOTO Statement کے دو سے زیادہ ٹرانسفر پوائنٹس ہو سکتے ہیں۔ پس یہ کثیر الشاخی سہولیات مہیا کر رہی ہوتی ہے۔

Syntax **سینٹیکس**

ON numeric variable or expression GOTO n1, n2, n3,...

(Interpretation) وضاحت

ایکسپریشن سے مراد درست BASIC ایکسپریشن ہے اور .... n1, n2, n3 سے مراد وہ لائنیں ہیں جہاں کنٹرول ٹرانسفر

ہوگا۔نومیرک ویری ایبل یا ایکسپریشن کی قیمت کی رینج یا ویری ایبل 255-0 ہے۔

اگر نومیرک ویری ایبل کی قیمت 1 ہے تو کنٹرول لائن n1 پر ٹرانسفر ہو جائے گا، اگر قیمت 2 ہے تو کنٹرول لائن n2 پر ٹرانسفر ہو جائے گا، اگر قیمت 3 ہے تو کنٹرول لائن n3 پر ٹرانسفر ہو جائے گا اور یہ سلسلہ اسی طرح چلتا رہے گا۔اگر قیمت GOTO سٹیٹمنٹ کے بعد کے لائن نمبرز سے کم یا زیادہ ہے تو کمپیوٹر "OUT OF RANGE" ایرر کا پیغام دے گا۔

مثال: آیئے ایک پروگرام پر غور کیجیے جس میں دو نمبرز A اور B کو جمع، تفریق، تقسیم اور ضرب کا بتلایا گیا ہے۔

```
5     INPUT A,B
10    INPUT "1-ADD, 2-SUB, 3-MUL, 4-DIV"; N
20    ON N GO TO 30,40,50,60
30    PRINT A+B : END
40    PRINT A-B : END
50    PRINT A*B : END
60    PRINT A/B : END
```

جب لائن 5 کو ایگزیکیوٹ کیا جاتا ہے تو سوالیہ نشان (؟) سکرین پر آتا ہے۔یہ دراصل ہے جہاں A اور B کی قیمتیں ٹائپ کی جاتی ہیں۔لائن نمبر 10 سکرین پر درج ذیل پیغام ظاہر کرتا ہے۔

1- ADD, 2- SUB, 3- MUL, 4- DIV ?

اگر آپ کی۔بورڈ کے ذریعہ 1 ان پٹ کرتے ہیں، تب N قیمت 1 لیتا ہے۔اگر آپ 2، 3 یا 4 انٹر کرتے ہیں تو N بالترتیب 2، 3 یا 4 قیمت لے گا۔اگر N کی قیمت

1 ہے تو کنٹرول لائن نمبر 30 پر ٹرانسفر ہو جاتا ہے،

2 ہے تو کنٹرول لائن نمبر 40 پر ٹرانسفر ہو جاتا ہے،

3 ہے تو کنٹرول لائن نمبر 50 پر ٹرانسفر ہو جاتا ہے،

4 ہے تو کنٹرول لائن نمبر 60 پر ٹرانسفر ہو جاتا ہے۔

لائن نمبر 30 کی ایگزیکیوشن پر سکرین پر A اور B کا حاصل جمع ظاہر ہوگا اور پروگرام ختم ہو جائے گا۔اسی طرح لائن نمبر 40، 50، 60 کے مطابق بالترتیب تفریق، ضرب اور تقسیم کے نتائج ظاہر ہوں گے۔

ایک پروگرام لکھنے کے کئی طریقے ہو سکتے ہیں۔آیئے مندرجہ بالا مثال میں دیئے گئے پروگرام کو ایک اور طریقے سے لکھتے ہیں۔

```
5     INPUT A, B
10    INPUT " 1-ADD, 2-SUB, 3-MUL, 4-DIV"; N
20    ON N GOTO 30, 40, 50, 60
30    PRINT A+B : GOTO 70
```

```
40      PRINT A-B  : GOTO 70
50      PRINT A*B : GOTO 70
60      PRINT A/B
70      END
```

ON ERROR GOTO Statement 3.2.2

اس کمانڈ سے GW-BASIC میں کسی ایرر کا پتہ چلانے میں مدد ملتی ہے اور ایرر ہینڈلنگ روٹین (Error handling-routine) کی پہلی لائن کی نشاندہی ہوتی ہے۔

Syntax **سینٹیکس**

ON ERROR GOTO *line number*

وضاحت (Interpretation)

لائن نمبر سے مراد پروگرام میں درست لائن نمبر ہے۔ GW-BASIC میں ایرر ہینڈلرز (Handlers) کو ON ERROR GOTO Statemet کے ذریعے آن کیا جاتا ہے۔ اس کو اکثر پروگرام کے شروع میں استعمال کیا جاتا ہے تاکہ پروگرام کے دوران کہیں پر موجود ایرر کو معلوم کیا جا سکے۔ جب پروگرام کی ایگزیکیوشن کے دوران کوئی ایرر واقع ہو تو کنٹرول فوری طور پر مخصوص لائن نمبر پر ٹرانسفر ہو جاتا ہے۔ اس لائن سے یوزر کے لیے بیان کی گئی ایرر ہینڈلنگ روٹین کام کرنا شروع کر دیتی ہے جو کہ ایرر کو مناسب انداز سے پروسیس کرتی ہے۔ اس کو براہِ راست یا بلاواسطہ موڈ میں ہینڈل (Handle) کیا جا سکتا ہے۔

GW-BASIC میں ہر ممکن ایرر کو ایک مخصوص کوڈ دیا گیا ہے۔ جب ایرر واقع ہوتی ہے تو اس کا کوڈ ایک خاص ویری ایبل جس کا نام ERR ہے کو دیا جاتا ہے اور لائن نمبر جہاں پر ایرر پکڑی گئی ہے کو ایک اور خاص ویری ایبل جس کا نام ERL ہے کو دیا جاتا ہے۔ ERR اور ERL مخصوص الفاظ ہیں۔

GW-BASIC کو یہ جاننے کی ضرورت ہوتی ہے کہ اس نے کب ایرر کو ہینڈل کرنا ختم کیا ہے۔ آپ RESUME, RESUME NEXT, RESUME line number یا END Statement کو استعمال کرتے ہوئے ایرر ہینڈلنگ روٹین سے باہر آ سکتے ہیں۔ RESUME Statement ایگزیکیوشن کو اُس سٹیٹمنٹ پر واپس لے جاتی ہے جو کہ ایرر کا باعث بنی ہے اور اسے دوبارہ ایگزیکیوٹ کرنے کی کوشش کرتی ہے۔ پس اگر پروگرام یا یوزر مسئلہ کو فوری طور پر حل کر سکتے ہوں تو ہمیں یہ سٹیٹمنٹ استعمال کرنی چاہیے۔ RESUME NEXT ایرر کا باعث بننے والی سٹیٹمنٹ کے فوراً بعد آنے والی سٹیٹمنٹ سے دوبارہ ایگزیکیوشن شروع کرتی ہے اور RESUME line number مخصوص لائن نمبر سے دوبارہ ایگزیکیوشن شروع کرتی ہے۔

مثال:

```
10      ON ERROR GOTO 70
20      INPUT "Enter first  No. ", n1%
30      INPUT "Enter second No.", n2%
40      r% = n1% * n2%
50      PRINT "Result ="; r%
```

```
60      END
70      PRINT "Error Code = "; ERR
80      PRINT "Error is on line no. "; ERL
90      END
RUN
Enter first No. 400
Enter second No. 450
Errer Code = 6
Errer is on line no. 40
```

یہاں r انٹیجر ویری ایبل ہے اور 450 * 400 کا نتیجہ اس میں ذخیرہ نہیں کیا جا سکتا۔ اس لیے اوورفلو (Overflow) ایرر واقع ہوتی ہے جس کا کوڈ 6 ہے اور جسے متغیر ERR کو دیا گیا ہے۔

3.3 سلیکشن سٹرکچر (Selection Structure)

سلیکشن سٹرکچر چناؤ کرتا ہے کہ کونسی متبادل پروگرام سٹیٹمنٹ کو ایگزیکیوٹ کرنا ہے۔ GW-BASIC میں سلیکشن سٹرکچر کو لاگو کرنے کے لیے ہمارے پاس IF...THEN اور IF...THEN...ELSE کی سٹیٹمنٹس ہیں۔

3.3.1 The IF THEN Statement

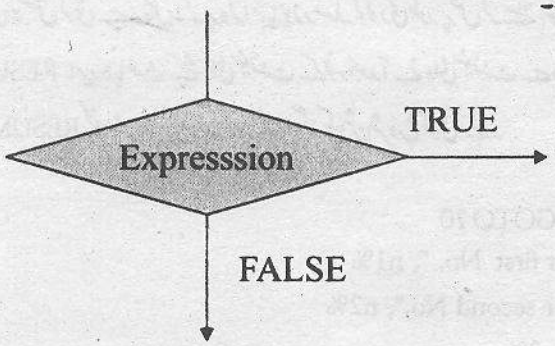
IF...THEN ایک فیصلہ کرنے والی سٹیٹمنٹ ہے جو کہ فیصلہ پر منحصر ہوتی ہے۔ یہ پروگرام کی ایگزیکیوشن کے آرڈر کو تبدیل کر سکتی ہے۔ یہ ایک پروگرام میں شرط پر مبنی پاتھ (Path) فلو کو منتخب کرنے میں استعمال ہوتی ہے۔ ''شرط'' ایک ایکسپریشن ہے جس کا جواب درست (جسے عام طور پر 1 سے ظاہر کیا جاتا ہے) یا غلط (جسے صفر سے ظاہر کیا جاتا ہے) کی شکل میں آتا ہے۔

Syntax **سینٹیکس**

☆ IF expression THEN Statement

☆ IF expression THEN line number

اگر ایکسپریشن درست ہے تب یا تو مخصوص لائن نمبر کی سٹیٹمنٹ یا THEN Keyword کے بعد آنے والی سٹیٹمنٹ ایگزیکیوٹ ہوتی ہے۔



شکل 3.1: IF...THEN سٹیٹمنٹ کا فلوچارٹ۔

مثال: کلومیٹرز میں طے کیے گئے سفر کے لیے گاڑی کا کرایہ معلوم کرنے کے لیے پروگرام تحریر کیجیے۔ کم سے کم وصول کیا گیا کرایہ 6 روپے ہے۔ یہ کم سے کم کرایہ 5 کلومیٹر یا اس سے کم سفر پر ہے۔ 5 کلومیٹر کے بعد 0.75 روپے فی کلومیٹر کے حساب سے کرایہ بڑھتا جاتا ہے۔

```
10      INPUT "KILOMETER", K
20      IF K < = 5 THEN PRINT " Rs. 6"; END
30      CHARGE! = 6+(K-5) * .75!
40      PRINT CHARGE!
50      END
```

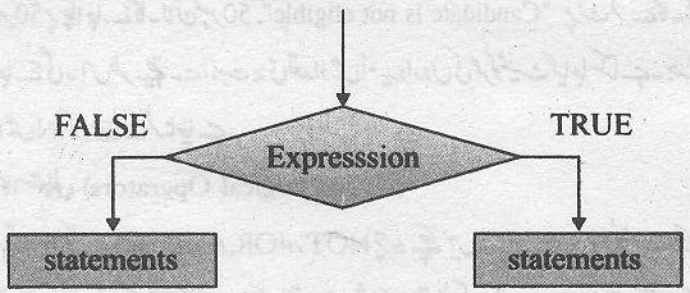
یہاں لائن نمبر 20 میں اگر K، 2 سے کم یا اس کے برابر ہے تب یہ 6 روپے پرنٹ کرے گا اور پروگرام ختم ہو جائے گا۔ لائن نمبر 20 کے تسلسل میں اگلی سٹیٹمنٹ END ہے۔ ورنہ اگر K، 2 سے چھوٹا یا اس کے برابر نہیں ہے تب لائن نمبر 30,40 اور 50 ایگزیکیوٹ ہوں گی۔

3.3.2 The IF...THEN...ELSE Statement

IF Statement, ELSE کی ورڈ کے ساتھ دو مختلف متبادل سٹیٹمنٹس کو مخصوص کرنے کے لیے استعمال ہوتا ہے۔ دی گئی متبادل سٹیٹمنٹس میں سے شرط کے مطابق ایک سٹیٹمنٹ پر عمل ہوتا ہے۔

Syntax سینٹیکس

IF (expression) THEN	**IF (expression) THEN line number**
Statements (true task)	
ELSE	**ELSE**
Statements (false task)	Statement (false task)



شکل 3.2: IF...THEN...ELSE سٹرکچر کا فلوچارٹ۔

IF...THEN...ELSE سٹیٹمنٹ ایک فیصلہ کرنے والی سٹیٹمنٹ ہے کیونکہ یہ پروگرام کے ہاتھ کا فیصلہ کرتی ہے۔ یہ موازنہ کرنے اور ٹیسٹ کرنے میں مدد دیتی ہے کہ کوئی شرط پوری ہوتی ہے یا نہیں۔ IF کے بعد درست BASIC شرط یا ایکسپریشن آتا ہے۔ اگر شرط دُرست پائی جائے تو THEN کے بعد والی لائن نمبر یا ہدایت پر عمل کیا جاتا ہے۔ بصورتِ دیگر ELSE کے بعد والی لائن نمبر یا ہدایت پر عمل کیا جاتا ہے۔

مثال: مختلف اُمیدوار جو کہ PIA کے سلیکشن بورڈ کے سامنے پیش ہو رہے ہیں کی عُمریں لی جاتی ہیں۔ اگر عُمر 17 سے کم ہے تو اُمیدوار مبینہ عہدے کے قابل نہیں ہے، بصورتِ دیگر وہ ٹیسٹ اور انٹرویو دے سکتا ہے۔ ہمیں اِس مسئلہ کے لیے ایک پروگرام لکھنے کو کہا جاتا ہے۔

```
10    INPUT "AGE"; A
20    IF A>= 17 THEN 30 ELSE 50
30    PRINT "Candidate is eligible"
40    GO TO 60
50    PRINT "Candidate is not eligible"
60    INPUT "World you like to input again (Y/N)"; Y$
70    IF Y$ = "Y" THEN 10
80    END
```

لائن نمبر 10، سکرین پر AGE? کا پیغام ظاہر کرے گی۔ آپ، فرض کریں کہ 18 کو بطور عُمر، کی بورڈ کے ذریعے اِن پُٹ کرتے ہیں۔ لائن نمبر 20 ٹیسٹ کرتی ہے کہ آیا کہ A>=17 یا نہیں۔ چونکہ A=18 جو کہ 17 سے بڑا عدد ہے، لائن نمبر 30 کی سٹیٹمنٹ ایگزیکیوٹ ہو جاتی ہے۔ لائن نمبر 30 "Candidate is eligible" پرنٹ کرتی ہے۔ لائن نمبر 40 کنٹرول لائن نمبر 60 کو منتقل کر دیتی ہے۔ لائن نمبر 60 پیغام "Would you like to input again (Y/N)?" پرنٹ کرتی ہے۔ ہم Y یا N اِن پُٹ کرتے ہیں۔ لائن نمبر 70 میں اگر اِن پُٹ Y ہو تو کنٹرول، لائن نمبر 10 کو چلا جاتا ہے۔ اگر اِن پُٹ N ہو تو کنٹرول لائن نمبر 80 پر چلا جاتا ہے یعنی پروگرام END ہو جاتا ہے۔

اَب اگر ہم لائن نمبر 60 میں Y کو اِن پُٹ کرتے ہیں تب کنٹرول دوبارہ لائن نمبر 10 کو منتقل ہو جائے گا۔ فرض کریں ہم ایک اور عمر 13 دیتے ہیں۔ لائن 20 میں A (یعنی 13) کی قیمت 17 سے زیادہ نہیں ہے۔ اِس لیے ELSE حصہ کو ایگزیکیوٹ کیا جائے گا اور کنٹرول لائن نمبر 50 پر چلا جائے گا۔ لائن نمبر 50 "Candidate is not eligible" پرنٹ کرے گا۔ پھر پہلے کی طرح لائن نمبر 60 ایگزیکیوٹ کی جائے گی۔ اِس طریقے سے، بہت بڑی تعداد میں اُمیدواروں کی عمر کو ٹیسٹ کیا جا سکتا ہے۔ جب ہم رکنا چاہیں تو ہمیں Y$ کی جگہ لائن 60 میں N کو اِن پُٹ کرنا چاہیے۔

لوجیکل آپریٹرز کا استعمال (Use of Logical Operators)

ہم تین لوجیکل پرومپٹز AND, OR اور NOT پڑھ چکے ہیں۔ یہ اوپریٹرز IF سٹیٹمنٹ کے ساتھ شرائط دینے میں اہم کردار ادا کرتے ہیں۔ اب تک ہم IF سٹیٹمنٹ کے ساتھ سادہ شرائط استعمال کرتے رہے ہیں۔ اِس سیکشن میں، ہم مشاہدہ کریں گے کہ کیسے پیچیدہ پروگرام لوجک (Logic) کو لوجیکل اوپریٹرز اِستعمال کرتے ہوئے سادہ بنایا جا سکتا ہے۔

مثال: آیئے دیے گئے تین اعداد میں سے سب سے چھوٹا عدد تلاش کرنے کے لیے ایک پروگرام پر غور کرتے ہیں۔

```
10    INPUT A,B,C
20    IF A<B AND A<C THEN 50 ELSE 30
30    IF B<A AND B<C THEN 60 ELSE 40
```

```
40      IF C<A AND C<B THEN 70
50      PRINT "A Is The Smallest Number": GO TO 80
60      PRINT "B Is The Smallest Number": GO TO 80
70      PRINT "C Is The Smallest Number"
80      INPUT "World you like to input again (Y/N)"; Y$
90      IF Y$ = "Y" THEN 10
100     END
```

آپ کو ایسی بہت سی حالتوں سے واسطہ پڑ سکتا ہے جہاں لوجیکل اوپریٹرز کا اِستعمال پروگرام لوجک کو سادہ بنائے گا۔ ہمیں صرف مدِنظر (پیشِ نظر) مسئلہ پر توجہ مرکوز رکھنا پڑتی ہے۔ اِس کتاب کے اگلے ابواب میں ہم لوجیکل اوپریٹرز کی مزید مثالیں دیکھیں گے۔

3.4 لُوپس (Loops)

ہمیں اکثر ایسے مسائل کا سامنا کرنا پڑتا ہے جن کے حل کے لیے سٹیٹمنٹس کے سیٹ کو بار بار ایگزیکیوٹ کرنے کی ضرورت ہو سکتی ہے۔ اِن حالات میں، ہمیں ایسے سٹرکچر کی ضرورت ہوتی ہے جو ہمیں سٹیٹمنٹس کے ایک سیٹ کی ایک حد تک یا کسی خاص شرط کے پورا ہونے تک دُہرانے کی اجازت دے سکتا ہے۔ لُوپ سٹرکچر اِس بنیادی ضرورت کو پورا کرتا ہے۔

3.4.1 FOR...NEXT LOOP

جب ہمیں قبل از وقت معلوم ہو کہ کتنی مرتبہ لُوپ کو دُہرانا چاہیے تو FOR...NEXT لوپ بہت زیادہ پُر اثر ہے۔ FORLOOP مخصوص تعداد میں سٹیٹمنٹس کے سیٹ کو دُہرانے کے لیے اِستعمال کیا جاتا ہے۔

Syntax **سینٹیکس**

```
FOR variable = x TO y [STEP z]
.
.
NEXT [variable]
```

وضاحت (Interpretation)

FOR کے بعد آنے والا نومیرک ویری ایبل نام لوپ کنٹرول ویری ایبل یا لوپ ویری ایبل کہلاتا ہے، x اور y نومیرک کانسٹینٹس ہیں جبکہ x لُوپ کی آغاز کی اور y آخری قیمت دیتا ہے۔ کی بورڈ STEP کے بعد z، x میں اضافہ کرتا ہے جب تک کہ y نہ آ جائے۔ اضافہ منفی بھی ہو سکتا ہے۔

y,x اور z نومیرک ویری ایبل نام بھی ہو سکتے ہیں۔ اِن حالات میں، لُوپ کو شروع کرنے سے پہلے اِن کی نومیرک قیمتیں دے دینی چاہئیں۔ یعنی FOR سٹیٹمنٹ سے پہلے کی ورڈ NEXT کے پاس ویسا ہی کنٹرول متغیر ہونا چاہیے جیسا کہ کی ورڈ FOR کے پاس ہے۔

مثال: آیئے ایک پروگرام پر غور کرتے ہیں جو کہ مندرجہ ذیل سیریز کے مجموعہ کو پرنٹ کرتا ہے۔

2,4,6,8,.....100

```
10    N = 0
20    FOR I = 2 TO 100 STEP 2
30    N = N+I
40    NEXT I
50    PRINT "SUM OF SERIES ="; N
60    END
```

3.4.2 WHILE...WEND Loop

"WHILE Loop" متعلقہ شرط کے غلط ہونے تک مطلوبہ عمل دہراتا رہتا ہے۔ یہ وہاں کارآمد ہوتا ہے جہاں پروگرام پہلے سے نہیں جانتا کہ لُوپ کو کتنی مرتبہ دہرایا جائے گا۔

Syntax سینٹیکس

```
WHILE expression
.
.
.
[Loop statements]
.
.
.
WEND
```

وضاحت (Interpretation)

"WHILE Loop" میں ایکسپریشن، لُوپ کے دُہرائے جانے کو کنٹرول کرتا ہے۔ جب دی گئی شرط دُرست ہو تو جن سٹیٹمنٹس کو ایگزیکیوٹ کیا جاتا ہے لوپ کی باڈی (Body) بناتی ہیں۔ شرط کے صحیح ہونے تک لُوپ کی باڈی (Body) ایگزیکیوٹ کی جاتی ہے۔ جیسے ہی یہ غلط ہوتی ہے لُوپ فوراً ختم ہو جاتا ہے اور پروگرام کنٹرول WEND سٹیٹمنٹ سے اگلی سٹیٹمنٹ کی طرف ٹرانسفر ہو جاتا ہے۔

یہ نوٹ کیا جانا چاہیے کہ WHILE لُوپ میں لُوپ کنٹرول ویری ایبل ہمیشہ لُوپ باڈی کے باہر سے شروع کیا جاتا ہے اور یہ لُوپ باڈی کے اندر، پروگرام لوجک کے مطابق زیادہ یا کم کیا جاتا ہے۔ جبکہ FOR لُوپ میں لُوپ کنٹرول متغیر FOR سٹیٹمنٹ کے اندر شروع، زیادہ یا کم کیا جاتا ہے۔

مثال: 1 سے 10 تک ہندسے پرنٹ کرنے کے لیے ایک پروگرام لکھیے۔

```
10      N = 1
20      WHILE N < =10
30      PRINT N
40      N = N+1
50      WEND
60      END
```

3.4.3 نیسٹڈ لوپ (Nested Loop)

ایک لوپ (FOR یا WHILE) کے اندر ایک یا ایک سے زیادہ (FOR یا WHILE) لوپس ہو سکتے ہیں۔ ایسے لوپ کو نیسٹڈ لوپ کہتے ہیں۔

مثال: فرض کریں کہ ہم آؤٹ پٹ کو مندرجہ ذیل طریقہ میں پرنٹ کرنا چاہتے ہیں۔

```
*****
****
***
**
*
```

```
10      FOR Y = 5 TO 1 STEP1
20      FOR X = 1 TO Y
30      PRINT "*";
40      NEXT X
50      PRINT
60      NEXT Y
70      END
```

یہ پروگرام دو لوپس پر مشتمل ہے: باہر والا لوپ لائن نمبر 10 سے 60 تک اور اندر والا لوپ لائن نمبر 20 سے 40 تک ہے۔ لائن نمبر 10 میں، ابتدائی طور پر Y کو 5 قیمت دی جاتی ہے۔ جب کہ Y کی قیمت 1 سے زیادہ ہوتی ہے، کنٹرول لائن نمبر 20 کو منتقل کیا جاتا ہے جس سے اندر والا لوپ 5 مرتبہ ایگزیکیوٹ ہوتا ہے۔ نتیجہ کے طور پر ایک قطار میں 5 سٹارز (*) چھپتے ہیں۔ لائن نمبر 50 پر سٹیٹمنٹ، پرنٹر کنٹرول کو اگلی لائن کے آغاز پر منتقل کرے گا۔ جب لائن نمبر 60 آتی ہے تو کنٹرول، واپس لائن نمبر 10 پر چلا جاتا ہے۔ اَب Y کی قیمت 4 ہو جاتی ہے اور ایک بار پھر اندر والا لوپ 4 مرتبہ ایگزیکیوٹ کیا جاتا ہے۔ نتیجہ کے طور پر دوسری قطار میں 4 سٹارز (*) پرنٹ ہوتے ہیں۔ یہ پروسیس اُس وقت تک جاری رہے گا جب تک Y کی قیمت 1 نہ ہو جائے۔ اِس کے بعد یہ ختم ہو جائے گا۔

مشق

1- خالی جگہیں پر کریں:

(i) ۔۔۔۔۔۔۔ سٹیٹمنٹ کے استعمال سے گریز کرنا چاہیے۔

(ii) GOTO سٹیٹمنٹ ۔۔۔۔۔۔۔ ٹرانسفر پوائنٹ مہیا کرتی ہے۔

(iii) FOR...NEXT لوپ ایک ۔۔۔۔۔۔۔ کی بنیاد پر کنٹرول ٹرانسفر کرتا ہے۔

(iv) ON...GOTO سٹیٹمنٹ کی مدد سے زیادہ سے زیادہ ۔۔۔۔۔۔۔ ٹرانسفر پوائنٹس مقرر کیے جاسکتے ہیں۔

(v) ON ERROR GOTO سٹیٹمنٹ GW-BASIC کی ایرراں پکڑنے کی صلاحیت ۔۔۔۔۔۔۔

(vi) GW-BASIC میں ہر ERROR کو ایک ۔۔۔۔۔۔۔ کوڈ دیا جاتا ہے۔

(vii) BASIC میں ۔۔۔۔۔۔۔ کے نام کے خاص ویری ایبل کو ایرر واقع ہونے پر ایرر کوڈ دیا جاتا ہے۔

(viii) BASIC پروگرام میں جہاں ایرر ہوتی ہے ۔۔۔۔۔۔۔ کے نام کے خاص ویری ایبل کو لائن نمبر دیا جاتا ہے۔

(ix) جب BASIC پروگرام میں کوئی ایرر واقع ہوتی ہے تو ۔۔۔۔۔۔۔ روٹین کو کنٹرول ٹرانسفر ہوسکتا ہے۔

(x) ۔۔۔۔۔۔۔ سٹرکچر عمل درآمد کے لیے پروگرام کی متبادل سٹیٹمنٹس سلیکٹ کرتی ہے۔

2- درست اوپشن منتخب کریں:

(i) ان میں سے کون سا لوجیکل اوپریٹر نہیں ہے؟

(a) AND (b) OR

(c) NEITHER (d) NOT

(ii) ایرر ہینڈلنگ روٹین سے نکلنے کے لیے ان میں سے کون سا اوپشن استعمال نہیں ہوسکتا؟

(a) RESUME (b) RESUME NEXT

(c) END (d) STOP

(iii) ان میں سے کون سی ملٹی برانچنگ (Multi branching) / کثیر الشاخی سٹیٹمنٹ ہے؟

(a) IF...ELSE (b) GOTO

(c) ON....GOTO (d) ON ERROR GOTO *line_number*

(iv) اگر ON...GOTO سٹیٹمنٹ میں کی ورڈ ON کے مطابق کسی نومیرک ایکسپریشن کی انٹیجر ویلیو 255 سے زیادہ ہو تو کس قسم کی ایرر واقع ہوتی ہے؟

(a) سنٹیکس ایرر (b) لوجیکل ایرر

(c) رن ٹائم (Run time) ایرر (d) یہ ایک ایرر نہیں ہے

(v) FOR...NEXT سٹیٹمنٹ استعمال کرتے ہیں:

(a) تکرار کے لیے (b) انتخاب کے لیے

(c) ترتیب کے لیے (d) ان میں سے سب

-3 صحیح اور غلط کی نشاندہی کیجیے۔

(i) RESUME سٹیٹمنٹ کے استعمال سے ایرر ہینڈلنگ روٹین سے نکلا جا سکتا ہے۔

(ii) END اور STOP سٹیٹمنٹس میں کوئی فرق نہیں ہے۔

(iii) غلط شرط کا نتیجہ NULL کی صورت میں نکلتا ہے۔

(iv) ایک WHILE...WEND لوپ تب تک کام کرتا ہے جب تک شرط درست رہتی ہے۔

(v) WHILE...WEND لوپ میں لوپ کنٹرول کرنے والا ویری ایبل ہمیشہ لوپ سے باہر ہوتا ہے۔

(vi) کسی لوپ کے اندر IF سٹیٹمنٹ دینے کے عمل کو نیسٹڈ لوپ کہتے ہیں۔

(vii) GOTO سٹیٹمنٹ پروگرام کی پیچیدگی بڑھا دیتی ہے۔

(viii) IF سٹیٹمنٹ میں دی گئی شرط نہ تو درست ہو سکتی ہے نہ ہی غلط۔

(ix) FOR...NEXT لوپ میں لوپ کنٹرول ویری ایبل کو اپ ڈیٹ کرنے کی ضرورت نہیں ہوتی، لوپ خود اسے ٹھیک کر لیتا ہے۔

(x) کسی اور IF سٹیٹمنٹ کے بلاک میں IF سٹیٹمنٹ دینے سے سینٹیکس ایرر واقع ہوتی ہے۔

-4 کنٹرول سٹرکچر کی تعریف بیان کریں۔ BASIC میں کتنی کنٹرول سٹرکچرز ہیں، مختصر طور پر بحث کریں۔

-5 نیسٹڈ لوپ کی تعریف بیان کریں۔ FOR...NEXT اور WHILE...WEND لوپ کا سنٹیکس لکھیں اور مثالوں سے وضاحت کریں۔

-6 کنٹرول کے ٹرانسفر سے کیا مراد ہے؟ بیسک میں مشروط اور غیر مشروط ٹرانسفر اور کنٹرول کے بارے میں بیان کریں۔

-7 WHILE...WEND اور FOR...NEXT لوپ میں فرق بیان کریں۔ ایسی صورتحال جس میں آپ لوپ چلانے سے پہلے تکرار کی تعداد سے آگاہ نہیں ہیں ان میں سے کون سا بہتر ہے؟

-8 مثلث کا رقبہ معلوم کرنے کے لیے پروگرام تحریر کریں۔ پروگرام یوزر سے قاعدہ اور ارتفاع کی قیمتیں لے گا اور رزلٹ بتائے گا۔

[اشارہ: ارتفاع × قاعدہ × $\frac{1}{2}$ = رقبہ]

-9 دائرے کا رقبہ اور محیط معلوم کرنے کے لیے پروگرام تحریر کریں۔ پروگرام یوزر سے دائرے کا رداس لے گا اور نتیجہ ظاہر کرے گا۔

[اشارہ: رداس × 3.14 × 2 = محیط اور رداس × رداس × 3.14 = رقبہ]

-10 WHILE...WEND لوپ کے استعمال سے پہلے دس طاق اعداد پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

-11 FOR...NEXT لوپ استعمال کرتے ہوئے پہلے پانچ جفت اعداد کے مربعوں کا حاصل جمع پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

-12 دو اعداد میں سے بڑا عدد معلوم کرنے کے لیے پروگرام تحریر کریں۔ پروگرام یوزر سے اعداد معلوم کرے گا۔

13- دیے گئے عدد کا ٹیبل پرنٹ کرنے کے لیے پروگرام تحریر کریں۔ پروگرام یوزر سے نمبر پوچھے گا۔

14- ایک پروگرام تحریر کریں جو طالب علم سے امتحان میں حاصل کر دہ نمبر معلوم کرے۔ اس کے بعد یہ فیصد معلوم کرے اور طالب علم کو گریڈ دے۔ گریڈ درج ذیل معیار کے مطابق دیا جائے گا۔

Percentage	Grade
>= 80	A1
>= 70, but < 80	A
>= 60, but < 70	B
>= 50, but < 60	C
>= 40, but < 50	D
< 40	F

جوابات

1- (i) GOTO (ii) ایک
(iii) شرط (iv) 255
(v) بحال کرتی ہے (vi) مخصوص
(vii) ERR (viii) ERL
(ix) ایرر ہینڈلنگ (x) سلیکشن

2- (i) c (ii) d
(iii) c (iv) c
(v) a

3- (i) صحیح (ii) غلط
(iii) غلط (iv) صحیح
(v) غلط (vi) غلط
(vii) صحیح (viii) غلط
(ix) صحیح (x) غلط

باب4

اریز

(Arrays)

4.1 تعارف (Introduction)

اس کتاب کے پچھلے باب میں ہم نے سادہ پروگرام لکھے ہیں جن میں کم تعداد میں متغیرات کو نومیرک یا سٹرنگ متغیرات کے طور پر استعمال کیا گیا ہے۔ اگر ہم نے ایک پروگرام میں سو یا ہزار مختلف متغیرات استعمال کرنے ہوں تو ہم ایسا کرنے کے قابل نہیں ہو سکتے۔ مثال کے طور پر ایک پروگرام کو طلبہ کے سو ناموں کو کمپیوٹر میں سٹور کرنا ہے۔ اِسے سو مختلف متغیرات کے ناموں کی ضرورت ہے۔ لہٰذا ایک ارے متغیر مختلف قیمتوں کو علیٰحدہ کرنے کے لیے کئی ناموں کو بیان کرتے ہوئے اس قسم کے مسئلہ کو حل کر سکتا ہے۔ اَریز ایک ہی قسم کے بہت بڑے ڈیٹا کو پروسیس کرنے کے لیے استعمال ہوتے ہیں۔

4.1.1 ارے کیا ہے؟ (What is an Array?)

ایک ارے متغیرات کا سیٹ ہے جو کہ ایک ہی قسم کے ڈیٹا کو سٹور کر سکتا ہے۔ ہر میموری لوکیشن میں ایک قیمت درج ہوتی ہے جو ارے کا رُکن کہلاتی ہے۔ ارے کو کمپیوٹر میموری میں یکے بعد دیگرے میموری لوکیشنز سے ظاہر کیا جاتا ہے۔ میموری لوکیشنز کو ارے کے ارکان کہتے ہیں۔ ہر ایک ارے کا ایک نام ہوتا ہے اور ارے کے ارکان تک رسائی اُن کی پوزیشن یا لوکیشن نمبر کے ذریعہ ہوتی ہے۔ اس پوزیشن نمبر کو انڈیکس (Index) یا سب سکرپٹ (Subscript) کہتے ہیں۔ انڈیکس قیمت کو ارے کے نام کے ساتھ بریکٹوں میں لکھا جاتا ہے۔ اگر کوئی اور قیمت نہ دی گئی ہو تو ارے کے پہلے رکن کی انڈیکس ویلیو (value) صفر ہوتی ہے اور ہر اگلے رکن کی انڈیکس ویلیو بڑھتی چلی جاتی ہے۔

Syntax **سینٹیکس**

array name (size of array)

ارے کے نام کے بعد چھوٹی بریکٹ میں ایک سب سکرپٹ ہوتا ہے۔ جہاں ارے ویری ایبل، ارے کا ویری ایبل نام ہو۔ وہاں سب سکرپٹ/انڈیکس قیمت اس میں موجود ڈیٹا آئیٹمز کی تعداد سے مخصوص ہوتی ہے۔ ارے میں ہر رُکن کو ایک مخصوص ذخیرہ لوکیشن دی جاتی ہے اور سب سکرپٹ یا انڈیکس قیمت کو خاص رکن کی پوزیشن کی شناخت کے لیے استعمال کیا جاتا ہے۔ سب سکرپٹ جو کہ ایک ایکسپریشن یا سادہ ویری ایبل ہو سکتا ہے ارے نام کے بعد چھوٹی بریکٹس میں بند ہوتا ہے۔ ہمارا صرف غیر سب سکرپٹڈ ویری ایبلز سے واسطہ رہا ہے جو کہ سادہ ویری ایبلز ہیں جو کہ صرف ایک قیمت ذخیرہ کرنے کے قابل ہیں۔ ارے کو سب سکرپٹڈ ویری ایبل کہتے ہیں کیونکہ جب ہمیں ایک خاص رکن تک رسائی کی ضرورت ہوتی ہے تو ہمیں اس رکن کی طرف اشارہ کرنے کے لیے ایک سب سکرپٹ استعمال کرنا چاہیے تا کہ ہم اسے دوسرے ارکان سے مختلف ظاہر کر سکیں۔ ایک ارے میں تمام ارے ویری ایبلز کا ایک ہی جیسا ویری ایبل نام ہوتا ہے

جو کہ ایک پوری ارے کا نام بھی کہلاتا ہے۔ ارے متغیر نومیرک یا سٹرنگ بھی ہوسکتا ہے۔ مثال کے طور پر سٹرنگ ارے N$ پانچ ارکان N$ (0), N$ (1),..., N$ (4) پر مشتمل ہے۔ اِسے درج ذیل طریقے سے ظاہر کیا جاسکتا ہے۔

N$(0)	Saleem
N$(1)	Asma
N$(2)	Majeed
N$(3)	Bushra
N$(4)	Mehmood

جب 0, 1, 2, 3, 4 سب سکرپٹس ہیں جو کہ ارے لوکیشن کی شناخت کے لیے استعمال ہوتے ہیں۔ N$(0),N$ (1),...,N$ (4) میموری ایڈریسز (Addresses) ہیں جہاں قیمتیں سٹور ہوں گی۔ LET Statement, READ Statement,..., DATA Statement, INPUT Statement سے سب سکرپٹڈ ویری ایبلز کو قیمتیں دی جاسکتی ہیں۔ سب سکرپٹ/انڈیکس قیمت کا سائز ایک متغیر، نمبر یا ایک نومیرک ایکسپریشن ہوسکتا ہے۔

4.1.2 ارے کو فل اور پرنٹ کرنا (Filling and Printing of an Array)

ارے میں ڈیٹا (سٹرنگ اور نمبرز)، INPUT Statements یا LET, READ سے اینٹر کیا جاتا ہے۔ ارے میں ڈیٹا سب سکرپٹڈ ویری ایبلز کو دیا جاسکتا ہے۔ درج ذیل پروگرام سٹرنگ کی قیمتوں کو ذخیرہ کرنے کے لیے ہے۔ اسے دونوں READ.... DATA Statement اور INPUT Statement سے ظاہر کیا گیا ہے۔

مثال 4.1۔

```
10      FOR K=1 TO 4
20      READ N$ (K)
30      PRINT N$ (K)
40      NEXT K
50      DATA Maheen, Misha, Mariam, Mahvash
60      END
RUN
Maheen
Misha
Mariam
Mahvash
```

جب اس پروگرام کو چلایا جاتا ہے تو سب سکرپٹڈ ویری ایبل N$(K) خود بخود ارے کے چار ارکان میں سے ہر ایک کو ظاہر کرتا ہے۔ پہلی مرتبہ لوپ کے ذریعے جبکہ K=1 سٹرنگ Maheen N$(1) کو دی جاتی ہے اور اسی طرح باقی بھی۔

مثال 4.2۔

```
10      FOR K=1 TO 4
20      INPUT "Enter the name of the student", N$(K)
30      NEXT K
40      FOR K=1 TO 4
50      PRINT N$ (K)
60      NEXT K
70      END
RUN
Enter the name of student      Maheen
Enter the name of student      Misha
Enter the name of student      Mariam
Enter the name of student      Mahvash
Maheen
Misha
Mariam
Mahvash
```

4.1.3 ڈم سٹیٹمنٹ (DIM Statement)

موجودہ سیٹنگ کے مطابق GW-BASIC میں ارے میں 10 ارکان ہوتے ہیں۔ جن کے سب سکرپٹس صفر تا 9 ہوتے ہیں۔ ہمیں 10 سے زیادہ سب سکرپٹ دینے کے لیے DIM سٹیٹمنٹ استعمال کرنا پڑتی ہے۔ مخصوص کیے گئے سب سے بڑے سب سکرپٹ سے بھی بڑا سکرپٹ استعمال کیا جائے تو ایک "Subscript out of range" کی ایرر واقع ہوتی ہے۔ ایک ارے کی زیادہ سے زیادہ سمتوں (dimensions) کی تعداد 255 ہوتی ہے۔

Syntax سینٹیکس

Line NO. DIM subscripted variable1, subscripted variable 2, ---

کی ورڈ DIM لفظ سمت (dimension) کی مختصر شکل ہے۔ درج بالا پروگرامز میں کوئی DIM Statement نہیں ہے کیونکہ یہ پروگرامز 11 سے کم قیمتوں کو ہینڈل کرتے ہیں۔

DIM کو میموری ویری ایبل بنانے کے لیے استعمال کیا جاتا ہے۔ یہ ارے ویری ایبل سکرپٹ کے لیے زیادہ سے زیادہ قیمت مخصوص کرتا ہے اور اسی طرح ڈیٹا سٹور کرنے کے لیے جگہ متعین کرتا ہے۔ جب سب سکرپٹ ویری ایبل کسی پروگرام میں استعمال ہوتے ہیں تو انہیں استعمال کرنے سے پہلے کچھ باتوں پر غور کرنا چاہیے۔

- ☆ سب سکرپٹ ویری ایبل کا نام
- ☆ سب سکرپٹ ویری ایبل کا سائز
- ☆ ہم ایک DIM سٹیٹمنٹ میں ایک سے زیادہ ارے متغیر کو بیان کر سکتے ہیں (جیسا کہ اوپر سینٹیکس میں بیان کیا گیا ہے)۔
- ☆ سب سکرپٹڈ ویری ایبلز سٹرنگ یا نومیرک ہو سکتے ہیں۔

درج ذیل پروگرام یوزر کی طرف سے دی گئی نمبروں کی فہرست میں سے سب سے بڑا عدد معلوم کرسکتا ہے۔

مثال 4.3۔

```
10      DIM NUM(100)
20      INPUT "How many numbers you want to enter; Max: 100:", LIMIT
30      FOR I= I TO LIMIT
40      INPUT "Enter any number", NUM(I)
50      NEXT I
60      LARGE = NUM(I)
70      FOR I = I TO LIMIT
80      IF LARGE < NUM(I) THEN LARGE = NUM(I)
90      NEXT I
100     PRINT "Largest number of list is"; LARGE
110     END
RUN
How many numbers you want to enter; Max: 100:14
Enter any number    56
Enter any number    88
Enter any number    2
Enter any number    69
Enter any number    5
Enter any number    14
Enter any number    34
Enter any number    55
Enter any number    76
Enter any number    54
Enter any number    35
Enter any number    29
Enter any number    81
Enter any number    7
Largest number of list is 88
```

4.2 ارے کی اقسام (Types of Array)

ارے کو دو بڑے حصوں میں تقسیم کیا جاسکتا ہے۔

☆ یک سمتی ارے (One-dimensional array) ☆ دو سمتی ارے (Two-dimensional array)

4.2.1 یک سمتی ارے (One-Dimensional Array)

یک سمتی ارے کو لینئر (Linear) ارے یا ویکٹر (Vector) ارے بھی کہتے ہیں۔ یہ صرف ایک قطار (Row) یا ایک کالم (Column) پر مشتمل ہوتا ہے۔ اسے 1-D ارے بھی کہتے ہیں۔ مثال کے طور پر 5 طلبہ پر مشتمل ایک کلاس نے ایک مضمون میں نمبر حاصل کیے ہیں اور ہم ہر ایک طالب علم کے اوسط (Average) نمبر معلوم کرنا چاہتے ہیں۔ مثال 4.4 میں دیا گیا پروگرام یہ معلوم کر سکتا ہے۔

یک سمتی ارے لکھنے کے لیے جنرل سینٹیکس یہ ہے۔

Line No DIM Array *name/variable(n)*

جہاں ارے (ویری ایبل) کا کام ارے ویری ایبل کے نام کو ظاہر کرتا ہے، n ارکان کے سائز کو ظاہر کرتا ہے۔

مثال 4.4۔

```
10    FOR I = 1 TO 5
20    READ MARK(I)
30    SUM=SUM + MARK(I)
40    NEXT I
50    AVG = SUM / 5
60    FOR I = 1 TO 5
70    PRINT MARK(I)
80    NEXT I
90    PRINT "SUM OF MARKS OF A CLASS IN A SUBJECT = "; SUM
100   PRINT "AVERAGE OF MARKS ="; AVG
110   DATA 88, 66,49,55,78
120   END
RUN
88
66
49
55
78
SUM OF MARKS OF A CLASS IN A SUBJECT = 336
AVERAGE OF MARKS  = 67.2
```

4.2.3 دو سمتی ارے (Two-Dimensional Array)

دو سمتی ارے قطاروں اور کالموں پر مشتمل ہوتا ہے۔ اسے جدول (table) یا قالب (matrix) بھی کہتے ہیں۔ دو سمتی ارے کو اس طرح بھی بیان کیا جا سکتا ہے: ''ایک سمتی اریز کا ارے''۔ دو سمتی ارے کے رکن کو دو انڈیکس قیمتوں سے ظاہر کیا جاتا ہے۔ ایک

انڈیکس قیمت قطار کو دوسری انڈیکس قیمت کالم کو ظاہر کرتی ہے۔ ایک ارے جسے ایک خاص رکن کی شناخت کے لیے دو سب سکرپٹس کی ضرورت ہوتی ہے کو ڈبل سب سکرپٹڈ ارے بھی کہتے ہیں۔ مثال کے طور پر اگر A ایک دو سمتی ارے ہے جس میں 4 قطاریں اور 3 کالم ہیں تو اس کا پہلا رکن A (0,0) اور آخری رکن A (3,2) ہے۔

دو سمتی ارے لکھنے کے لیے جنرل سینٹیکس یہ ہے۔

Line No DIM *array variable* (row, col)

جب کہ "ارے ویری ایبل" دو سمتی ارے کے نام کو ظاہر کرتا ہے۔ قطار (row) جدول میں موجود تمام قطاروں کو ظاہر کرتی ہے۔ یہ ایک غیر علامتی نمبر ہے۔ کالم (col) جدول میں موجود تمام کالموں کو ظاہر کرتا ہے۔ مثال کے طور یہ بتلانے کے لیے کہ A میں 4 قطاریں اور 3 کالم ہیں ہم اسے اس طرح لکھیں گے A (3,2)۔ درج ذیل جدول میں "A" کے تمام ارکان کی تعداد 12=3×4 ہے۔

R/C	0	1	2
0	A (0,0)	A (0,1)	A (0,2)
1	A (1,0)	A (1,1)	A (1,2)
2	A (2,0)	A (2,1)	A (2,2)
3	A (3,0)	A (3,1)	A (3,2)

درج بالا جدول میں چار قطاریں اور تین کالم ہیں۔ اس کو دو سمتی ارے میں پڑھا اور پروسیس کیا جا سکتا ہے۔

4.2.3.1 دو سمتی ارے کی فلنگ اور پرنٹنگ (Filling and Printing of Two-Dimensional Arrays)

دو سمتی ارے کے انفرادی ارکان میں ڈیٹا انٹر کیا جاتا ہے۔ ڈیٹا انٹر کرنے کے لیے رکن کا حوالہ انڈیکس یا سب سکرپٹ سے دیا جاتا ہے۔ اسی طرح، ایک ارے سے ڈیٹا ارے کے انفرادی ارکان سے حاصل کیا جاتا ہے۔ عام طور پر نیسٹڈ لوپس دو سمتی ارے کے ارکان تک رسائی کے لیے استعمال کیے جاتے ہیں۔ درج ذیل مثال میں دوارے کو ڈیٹا دیا جاتا ہے اور پھر کمپیوٹر سکرین پر اسی ڈیٹا کو ٹیبل کی شکل میں پرنٹ کیا جاتا ہے۔

جدول کے ارکان میں ڈیٹا انٹر کرنے کے لیے نیسٹیڈ لوپ کو استعمال کیا گیا ہے۔ باہر کے لُوپ کو قطاروں کی انڈیکس قیمت تبدیل کرنے کے لیے استعمال کیا گیا ہے اور اندر کے لُوپ (Inner Loop) کو کالموں کی انڈیکس قیمت تبدیل کرنے کے لیے استعمال کیا گیا ہے۔ اسی طرح کا طریقہ جدول کے ارکان سے ڈیٹا پرنٹ کرنے کے لیے استعمال کیا جاتا ہے۔

مثال 4.6۔

```
10    DIM A(2,2), B(2,2), Z(2,2)
20    FOR R = 1 TO 2
30    FOR C = 1 TO 2
40    READ A(R,C), B(R,C)
```

```
50      Z(R,C) = A(R,C) + B(R,C)
60      PRINT  Z(R,C),
70      NEXT C
80      PRINT
90      NEXT R
100  DATA  8,4,3,5
110  DATA  6,4,5,5
120  END
RUN
12      8
10      10
```

ارے کی مینی پولیشن (استعمال) کا طریقہ (Array Manipulation)

ارے کو استعمال کرتے ہوئے مختلف کام (operations) کیے جاسکتے ہیں۔ جیسا کہ ارے میں ایک خاص رکن کو ڈھونڈنا، دو مختلف اریز سے ارکان کا موازنہ (matching)، ارے کو سٹور کرنا، ارے سے چھوٹے سے چھوٹا اور بڑے سے بڑا عدد معلوم کرنا، اور ارے کو دوبارہ ترتیب دینا (سارٹنگ -Sorting)۔

مشق

-1 خالی جگہ پُر کیجیے۔

(i) دو سمتی ارے کو --------- بھی کہا جاتا ہے۔

(ii) ارے کی فہرست کو دوبارہ ترتیب دینا --------- کہلاتا ہے۔

(iii) DIM Statement اختیاری (optianal) ہوتی ہے اگر میموری لوکیشن کی تعداد ---------- سے کم ہو۔

(iv) ارے ---------- متغیرات کا مجموعہ کہلاتا ہے۔

(v) دو سمتی ارے میں سب سکرپٹس کو ---------- ذریعے علیٰحدہ کیا جاتا ہے۔

(vi) کسی سب سکرپٹ کی کم سے کم قیمت ہمیشہ ---------- قیاس کی جاتی ہے۔

(vii) Z(2,2) ---------- سمتی ارے کی مثال ہے۔

(viii) ایک سٹرنگ ارے اس کے نام کے ساتھ ---------- کی علامت لگا کر بیان کرتے ہیں۔

(ix) ایسے سب سکرپٹڈ ویری ایبلز کا مجموعہ جن کا نام ایک ہی ہو ---------- کہلاتا ہے۔

(x) ایک ارے کی بریکٹوں میں دی گئی قیمتیں ---------- کہلاتی ہیں۔

2- درست جواب کا انتخاب کیجئے۔

(i) ارے کی ---------اقسام ہیں۔

(a) 1 (b) 2 (c) 3 (d) 4

(ii) سٹیٹمنٹ X(30)--------- میموری لوکیشنز کو محفوظ کرتی ہے۔

(a) 29 (b) 30 (c) 31 (d) ان میں کوئی بھی نہیں

(iii) دو سمتی ارے میں اگر سمت نہ دی گئی ہو تو ارے میں ---------سے زیادہ ارکان نہیں ہونے چاہئیں۔

(a) 10 (b) 100 (c) 110 (d) 121

(iv) ارے سے بڑی سے بڑی قیمت حاصل کرنے کے لیے درج ذیل میں سے کون سی سٹیٹمنٹ استعمال کی جاتی ہے؟

(a) INPUT (b) READ....DATA

(c) ON-ERROR-GOTO (d) ان میں سے کوئی نہیں

(v) ان میں سے کون سا درست سب سکرپٹ نہیں ہے؟

(a) NUM(10) (b) A(2) (c) B(4) (d) A(-2)

(vi) ایک ارے کے رُکن کو ظاہر کرتے ہیں اس کے

(a) سب سکرپٹ سے (b) ارے سے (c) اوبجیکٹ سے (d) رکن کے نام سے

(vii) سمتی سٹیٹمنٹ کے لیے کی ورڈ ---------استعمال ہوتا ہے۔

(a) DMS (b) DS (c) DIM (d) DM

(viii) ایک سمت کے ارکان کی زیادہ سے زیادہ تعداد ہے

(a) 10 (b) 255 (c) 32767 (d) ان میں سے کوئی نہیں

(ix) دو سب سکرپٹڈ ویری ایبل p(3,2)--------- میں موجود ڈیٹا کے رُکن کو بیان کرتا ہے۔

(a) کالم تیسرا اور قطار دوسری (b) کالم تیسرا اور کالم دوسرا

(c) کالم دوسرا اور قطار تیسری (d) قطار تیسری اور قطار دوسری

(x) سٹیٹمنٹ DIM C(30,50) محفوظ کرتی ہے۔

(a) 80 مقامات (b) 1500 مقامات

(c) 1800 مقامات (d) 150 مقامات

3- صحیح یا غلط کی نشاندہی کیجیے۔

(i) ارے سب سکرپٹڈ متغیرات کا مختلف متغیر ناموں کے ساتھ اکٹھ ہے۔

(ii) ارے مختلف متغیرات کا سیٹ ہے۔

(iii) فہرست میں آئیٹمز کو ایک متغیر کے طور پر ظاہر کیا جاتا ہے جس کا نام ارے ہوتا ہے۔

(iv) پروگرامنگ میں فہرستوں یا جدول کو ارے کہتے ہیں۔

(v) DIM P(4,3) سٹیٹمنٹ کے لیے 12 میموری سپیسز محفوظ کیے جاتے ہیں۔

(vi) DIM ایک مخصوص (محفوظ) لفظ ہے۔

(vii) ایک سمتی ارے کو ٹیبل بھی کہتے ہیں۔

(viii) ایک بات جب کسی ارے کی سمت متعین کر دی جاتی ہے تو پروگرام میں نئے سرے سے اُس کی سمت متعین نہیں کی جا سکتی۔

(ix) وہ سب سے بڑا عدد جو بیسک اپنے آپ سب سکرپٹ کے طور پر مقرر کرتی ہے 100 ہے۔

(x) دو سمتی اریز کے نام اسی طرح رکھے جاتے ہیں جیسے ایک سمتی اریز کے۔

-4 DIM Statement سے کیا مراد ہے؟

-5 ارے میں سب سکرپٹ ویری ایبل کا استعمال بیان کیجیے۔

-6 آپ ارے کو کیسے فِل اور پرنٹ کریں گے؟

-7 ارے کی مینی پولیشن (Manipulation) سے کیا مراد ہے؟

-8 1-D ارے اور 2-D ارے میں فرق بیان کیجیے۔

-9 ایک مثال کی مدد سے دو سمتی ارے پرنٹ کرنے کا طریقہ بیان کریں۔

-10 انٹیجر قسم کے ڈیٹا کو ارے میں اینٹر کرنے کے لیے اور قیمتوں کو آؤٹ آرڈر میں پرنٹ کرنے کے لیے BASIC میں ایک پروگرام لکھیے۔

-11 سادہ اور سب سکرپٹڈ ویری ایبل میں فرق بیان کریں۔

-12 سوال نمبر 16 کے پروگرام کے لیے فلوچارٹ بنائیں۔

-13 ارے A کے ارکان اور ارے B کے ارکان کو جمع کرنے کے لیے الگورتھم لکھیں۔

-14 دیے گئے اعداد میں سے طاق اعداد کی فہرست پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

6, 42, 4, 77, 32, 9, 21, 22, 8, 45, 15, 46

-15 ایک پروگرام لکھیں جو بیس اعداد پر مشتمل ارے N پڑھے اور ارے کے ارکان کا حاصل ضرب معلوم کرے۔

-16 ایک پروگرام لکھیں جو یوزر کے دیے گئے بارہ اعداد پر مشتمل ارے Z پڑھے اور پھر ارے کے تمام ارکان کا مجموعہ اور اوسط پرنٹ کرے۔

-17 درج ذیل دیے گئے بیانات میں اگر ایررز موجود ہیں تو تلاش کریں۔

(a)
```
10  DIM N$(10)
20  FOR K= 4 TO 15
30  INPUT N$
40  NEXT I
```

(b)
```
10    FOR J = K TO 15
20    K(J) = J
30    PRINT K(J)
40    NEXT J
```

18- 20 ناموں کی فہرست کو ترتیب نزولی میں ترتیب دینے کے لیے پروگرام تحریر کریں۔

جوابات

1-	(i) ٹیبل	(ii) سارٹنگ	(iii) 11
	(iv) سب سکرپٹڈ	(v) کوما(,)	(vi) 0
	(vii) 2	(viii) $	(ix) ارے
	(x) سب سکرپٹس		
2-	(i) b	(ii) c	(iii) a
	(iv) d	(v) d	(vi) a
	(vii) c	(viii) b	(ix) c
	(x) b		
3-	(i) غلط	(ii) غلط	(iii) صحیح
	(iv) صحیح	(v) غلط	(vi) صحیح
	(vii) غلط	(viii) غلط	(ix) غلط
	(x) صحیح		

باب 5

سب پروگرامز اور فائل ہینڈلنگ

(Subprograms and File Handling)

5.1 تعارف (Introduction)

زیادہ تر کمپیوٹر لینگوئجز میں پروگرام زیادہ لمبا ہو جائے تو اس سے کام لینا زیادہ مشکل ہو جاتا ہے۔ سوائے GW Basic کے جو ان کا استعمال آسان بناتی ہے۔ ایک الگ بڑے پروگرام کو چھوٹے اور مینیج ایبل (Manageable) حصوں میں بانٹ دیا جاتا ہے، جنہیں سب پروگرام یا ماڈیولز (Modules) کہا جاتا ہے۔ اِسے اس طرح بنایا جاتا ہے کہ وہ ایک مخصوص کام کر کے خاص قیمت واپس کرتا ہے۔

بیسک میں دو طرح کے سب پروگرامز ہوتے ہیں: بلٹ-اِن (Built-in) اور یوزر ڈیفائنڈ فنکشنز (User defined functions) بیسک لینگوئج بلٹ-اِن فنکشنز مہیا کرتی ہے اور پروگرامر انہیں استعمال کر سکتا ہے تاکہ اسے بعض مواقع پر کوڈ نہ لکھنا پڑیں۔ کوئی خاص کام سرانجام دینے کے لیے پروگرامر یوزر ڈیفائنڈ فنکشنز لکھتا ہے۔ فنکشن ہمیشہ کالنگ (Calling) ماڈیول کو ایک خاص قیمت دیتا ہے۔ آیئے ہم سٹینڈرڈ (Standard) فنکشنز سے آغاز کرتے ہیں۔

جیسا کہ پہلے بتایا گیا ہے کہ سٹینڈرڈ فنکشنز بیسک لینگوئج کے ساتھ مہیا کیے جاتے ہیں اور استعمال کے لیے ان کا نام استعمال کرنا پڑتا ہے۔ ہر فنکشن کا ایک مخصوص نام ہوتا ہے اور اس کے بعد بریکٹیں دی جاتی ہیں۔ بریکٹوں میں آرگیومنٹ (Argument) کانسٹینٹ، ویری ایبل، ایکسپریشن، یا دوسرے فنکشن کی شکل میں دیا جاتا ہے۔ فنکشن آرگیومنٹ کے مطابق کام کرتا ہے۔ اگر کسی سٹیٹمنٹ میں کوئی فنکشن سب سے زیادہ ترجیحی بنیاد پر استعمال کیا جائے تو یہ سٹیٹمنٹ میں کسی ویلیو کی نسبت اس پر سب سے پہلے عمل کیا جائے گا۔

5.1.1 بلٹ اِن فنکشن (Built-in Function)

یہ فنکشنز دی گئی قیمتوں (operands) پر عوامل (operations) سرانجام دیتے ہیں اور نتائج مہیا کرتے ہیں۔ اصل میں یہ ایسے پروگرامز ہوتے ہیں جو لینگوئج بنانے والے لکھ کر اس میں شامل کر دیتے ہیں۔ ایسے فنکشنز کو سٹینڈرڈ فنکشنز یا لائبریری فنکشنز یا انٹرنزک (Intrinsic) فنکشن یا بلٹ اِن فنکشن کہا جاتا ہے۔ بیسک لینگوئج میں بہت سے پری-ڈیفائنڈ (Pre-defined) فنکشنز ہوتے ہیں جنہیں صرف ''کال (call)'' کر کے استعمال کیا جا سکتا ہے۔ بیسک بلٹ-ان فنکشنز کو دو حصوں میں تقسیم کیا جا سکتا ہے، نومیرک فنکشنز اور سٹرنگ فنکشنز۔

نومیرک فنکشنز (Numeric Functions)

ایسے فنکشنز کا صرف نومیرک ویلیوز پر ہی اطلاق کیا جا سکتا ہے اور ان کی مدد سے نومیرک نتائج حاصل کیے جا سکتے ہیں۔ بیسک میں بہت سارے فنکشنز موجود ہیں لیکن ہم صرف زیادہ اہم فنکشنز کے بارے میں پڑھیں گے۔

1۔ ABS فنکشن (ABS Function)

مقصد: ABS فنکشن کی مدد سے ایک ایکسپریشن کی مطلق قیمت حاصل ہوتی ہے یعنی ایسی قیمت جس کی کوئی منفی علامت نہ ہو۔

فارمیٹ (Format): ABS(x)

مثال:

```
10      PRINT ABS (-15)
20      PRINT ABS (-12.45)
RUN
15
12.45
```

2۔ INT فنکشن (INT Function)

مقصد: یہ فنکشن سب سے بڑا صحیح عدد (Integer) دیتا ہے جو کہ x سے چھوٹا یا برابر ہوتا ہے۔ مکمل عدد کی صورت میں اس کا جواب اسی عدد کی صورت میں آتا ہے۔

فارمیٹ: INT (x)

مثال:

```
10      J = INT (3.9999)
20      PRINT J
RUN
3
```

3۔ SQR فنکشن (SQR Function)

مقصد: یہ عدد x کا جذر دیتا ہے۔ x لازمی طور پر 0 سے بڑا یا برابر ہونا چاہیئے۔

فارمیٹ: SQR(n) جبکہ n = >0

فارمیٹ: SQR(x) جبکہ x = >0

مثال:

```
10      FOR X = 10 TO 25 STEP 5
20      PRINT X; SQR (X)
30      NEXT X
RUN
10      3.162278
15      3.872984
20      4.472136
25      5
```

4- SIN فنکشن (SIN Function)

مقصد: SIN فنکشن کا مقصد زاویہ x کی ریڈیئنز (Radians) / سطحی زاویوں میں ٹرگنومیٹرک (Trigonometric) نسبت معلوم کرنا ہے۔ ڈگری میں دیے گئے زاویے کو ریڈیئنز میں تبدیل کرنے کے لیے درج ذیل فارمولا استعمال کیا جاتا ہے۔

Angle in Radians = Angle in degree * π/180

SIN (x* π/180)

SIN (x) is calculation in single-precision.

فارمیٹ: SIN(x)

مثال:

```
10      PI= 3.142857
20      PRINT SIN (PI * 30/180)
.
.
Run
0.9999
```

اسی طرح دیگر ٹرگنومیٹرک فنکشنز بھی استعمال ہوتے ہیں جن کی فہرست درج ذیل ہے:

فنکشن	بیسک میں اس کا متبادل
Cosine	COS(x)
Secant	SEC(x)
Cosecant	COSEC(x)
Tangent	TAN(x)
Cotangent	COT(x)

5- FIX فنکشن (FIX Function)

مقصد: FIX فنکشن کا مقصد اعشاریہ والے حصہ کو چھوڑ کر باقی صحیح عدد حاصل کرنا ہے۔ FIX نمبر کو راؤنڈ آف نہیں کرتا۔

فارمیٹ: FIX(x)

مثال:

```
10      PRINT FIX (-7.09)
RUN
- 7
```

6- TAB فنکشن (TAB Function)

مقصد: سکرین (Screen) پر کسی خاص کالم x پر پرنٹ کرنے کے لیے۔

اگر موجودہ پرنٹنگ پوزیشن سپیس (Space) x چھوڑ کر ہو تو کرسر TAB سے اگلی لائن پر دی گئی پوزیشن پر چلا جاتا ہے۔ سپیس x سے مراد سب سے بائیں طرف کی پوزیشن ہے۔ سب سے دائیں طرف کی پوزیشن سکرین کی چوڑائی کے مطابق ہوتی ہے۔ x کی قیمت 1 تا 255 کے درمیان سے ہونی چاہیے۔

فارمیٹ: TAB (x)

مثال:

```
10      PRINT "PAKISTAN" TAB(2) "IS MY" TAB(4) "COUNTRY"
RUN
PAKISTAN    IS  MY          COUNTRY
```

7- RND فنکشن (RND Function)

مقصد: RND فنکشن 0 اور دیئے گئے نمبر کے درمیان ایک رینڈم (Random) ویری ایبل دیتا ہے۔

فارمیٹ: RND [(x)]

مثال:

```
10      FOR I = 1 to 5
20      PRINT INT (RND * 101)
30      NEXT I
40      END
RUN
53      30      31      51      5
```

8- LOG فنکشن (LOG Function)

مقصد: کسی نمبر کا قدرتی لوگرتھم معلوم کرنے کے لیے استعمال کرتے ہیں (بیسک میں، لوگرتھم کا بیس 2.718282=e ہوتا ہے)۔

فارمیٹ: LOG(x)

مثال:

```
10      PRINT LOG (10)
RUN
2.302585
```

9- SPC فنکشن (SPC Function)

مقصد: SPC فنکشن ایک پرنٹ سٹیٹمنٹ میں x سپیسز چھوڑنے کے لیے استعمال ہوتا ہے۔ SPC کو Print اور LPrint سٹیٹمنٹ

کے ساتھ استعمال کیا جاسکتا ہے۔اس کا آرگیومنٹ x یقینی طور پر 0 سے 255 کی رینج میں ہونا چاہیے۔اگر x کی ویلیو پرنٹر یا سکرین کی مقرر کردہ چوڑائی سے زیادہ ہو تو چوڑائی کی ویلیو nMOD استعمال ہوتی ہے۔

فارمیٹ: SPC (x)

مثال:

```
10      PRINT "OVER" SPC(15) "THERE"
Run
        1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
OVER                    THERE
```

10۔ BEEP فنکشن (BEEP Function)

مقصد: ایک چوتھائی سیکنڈ کے لیے 800 Hz (800 سائیکلز فی سیکنڈ) آواز پیدا کرنے کے لیے۔

فارمیٹ: BEEP

مثال: کمانڈ لیول پر ٹائپ کریں:

```
BEEP
```

11۔ DATE$ فنکشن (DATE$ Function)

مقصد: یہ فنکشن موجودہ تاریخ سیٹ کرنے کے لیے یا دیکھنے کے لیے استعمال ہوتا ہے۔

فارمیٹ: DATE$ = v$ یا v$ = DATE$ جبکہ v$ ایک درست سٹرنگ لٹرل یا ویری ایبل ہے۔تاریخ دیتے وقت v$ درج ذیل میں سے کسی ایک شکل میں ہوسکتا ہے۔

```
Ok
PRINT DATE$
11-14-2005
Ok

DATE$ = "12-03-2005"
Ok

PRINT DATE$
12-03-2005
Ok
```

mm-dd-yy
mm/dd/yy
mm-dd-yyyy
mm/dd/yyyy

مثال: شکل 2.7 DATE$ سٹیٹمنٹ کا استعمال دکھاتی ہے۔

سٹرنگ فنکشنز (String Function)

سٹرنگ فنکشنز کریکٹر، سٹرنگز کو پروسیس کرنے کے لیے استعمال ہوتے ہیں اور ان کی مدد سے نومیرک ویلیوز یا سٹرنگ ویلیوز کی شکل میں جواب ملتا ہے۔

1۔ LEN فنکشن (LEN Function)

مقصد: LEN فنکشن کی مدد سے ہم سٹرنگ x$ کے کریکٹرز کو گن سکتے ہیں۔x$ سے مراد کوئی سٹرنگ ایکسپریشن ہے۔اس میں خالی جگہ یا نہ پرنٹ ہونے والے کریکٹرز کو بھی گنا جاتا ہے۔

فارمیٹ: LEN (x$)

مثال:

```
10      A$ = "PAKISTAN"
20      PRINT LEN (A$)
RUN
8
```

2- VAL فنکشن (VAL Function)

مقصد: اس کی مدد سے ہمیں کسی سٹرنگ x$ کی نومیرک ویلیو ملتی ہے۔ یہ فنکشن آرگیومنٹ سٹرنگ (سٹرنگ کے ساتھ دیے گئے لفظ، جملے یا عبارت) میں شامل خالی جگہیں، ٹیب سے دیے گئے فاصلے اور لائن فیڈز (Line feeds) (اگلی لائن میں جانے کے لیے کمپیوٹر یا پرنٹر کو ہدایت دینے والے کنٹرول کریکٹرز) کو شمار نہیں کرتا۔ اگر سٹرنگ x$ کا پہلا کریکٹر نومیرک نہ ہو تو VAL(x$) فنکشن کا جواب صفر میں ہوگا۔

مثال:

```
10      PRINT VAL("78, city Lahore")
RUN
78
```

اس مثال میں VAL فنکشن ایڈریس میں سے نمبر معلوم کرنے کے لیے استعمال کیا گیا ہے۔

3- MID فنکشن (MID Function)

مقصد: MID فنکشن کسی بھی سٹرنگ ایکسپریشن میں سے مطلوبہ حصہ معلوم کرنے کے لیے استعمال کیا جاتا ہے۔ اس میں n ایک انٹیجر ہے جس کی ویلیو 1 سے 255 کے درمیان ہوتی ہے۔ m ایک اور انٹیجر ایکسپریشن ہے جس کی رینج 0 سے 255 ہے۔ یہ فنکشن سٹرنگ x$ سے، مطلوبہ کریکٹر (nth کریکٹرز) سے شروع ہونے والے m کریکٹرز کی لمبائی کا سٹرنگ دیتا ہے۔ اگر m کو چھوڑ دیا جائے یا اس سے کم کریکٹرز n کی دائیں جانب ہوں تو n کے دائیں طرف کے تمام کریکٹرز ہمیں ملتے ہیں۔ اگر m کی ویلیو 0 ہو یا اگر n، LEN(x$) سے بڑا ہو تو MID$ فنکشن کوئی سٹرنگ نہیں دیتا۔

فارمیٹ: V$ = MID$(x$,n[,m])

مثال:

```
10      A$ = "WE LOVE PAKISTAN"
20      PRINT MID$(A$, 4, 4)
30      PRINT MID$(A$, 9, 8)
RUN
LOVE
PAKISTAN
```

4- SPACES فنکشن (SPACES Function)

مقصد: اس کی مدد سے ہمیں x سپیسز (Spaces) کا سٹرنگ ملتا ہے۔ اس میں x ایک انٹیجر ہوتا ہے اور اسکی رینج 0 سے 255 ہوتی ہے۔

فارمیٹ: SPACES$(x)

مثال:

```
10      FOR N 1 TO 5
20      X$ = SPACE$(N)
30      PRINT X$ ; N
40      NEXT N
RUN
        1
         2
          3
           4
            5
```

پروگرام ایک لوپ (Loop) پر مشتمل ہے۔ جتنی بار لوپ چلتا ہے 20 نمبر لائن ایک سپیس کا اضافہ کر دیتی ہے۔

5- RIGHT$ فنکشن (RIGHT$ Function)

مقصد: RIGHT$ فنکشن کا مقصد سٹرنگ x$ کے سب سے دائیں طرف کے کریکٹرز مہیا کرنا ہے۔ اگر i کی ویلیو LEN(x$) سے بڑی یا برابر ہو تو RIGHT$ ہمیں x$ دیتا ہے۔ اگر i صفر کے برابر ہو تو کوئی سٹرنگ پرنٹ نہیں ہوتا۔

فارمیٹ: RIGHT$(x$,i)

مثال:

```
10      A$ =   "DISK OPERATOR"
20      PRINT  RIGHT$(A$,5)
RUN
RATOR
```

6- LEFT$ فنکشن (LEFT$ Function)

مقصد: یہ فنکشن سٹرنگ x$ کے سب سے بائیں طرف والے n کریکٹرز پرنٹ کرنے کے لیے استعمال ہوتا ہے۔ n کی رینج 0 سے 255 تک ہونی چاہیے۔ اگر n کی ویلیو LEN$(x$) سے زیادہ ہو تو تمام x$ پرنٹ ہو جاتا ہے۔ اگر n صفر کے برابر ہو تو کچھ بھی پرنٹ نہیں ہوتا۔

فارمیٹ: LEFT(x$, n)

مثال:

```
10      A$ = "DISK OPERATOR"
20      PRINT LEFT$(A$, 4)
RUN
DISK
```

7- CHR$ فنکشن (CHR$ Function)

مقصد: ASCII کوڈ کو اس کے متبادل کریکٹر میں تبدیل کرنے کے لیے استعمال ہوتا ہے۔ CHR فنکشن، ASC فنکشن کے اُلٹ کام کرتا ہے اور ASCII کنٹرول کریکٹرز حاصل کرنے کے کام آتا ہے۔ ان کریکٹرز کو پرنٹ کرنے کے لیے PRINT سٹیٹمنٹ استعمال کرتے ہیں۔

فارمیٹ: CHR$(n)

مثال:

```
10      PRINT CHR$(65)
RUN
A
```

5.1.2 یوزر۔ڈیفائنڈ فنکشنز (User Defined Functions)

ایک پروگرامر کوئی مسئلہ حل کرنے کے لیے یوزر ڈیفائنڈ فنکشن بناتا ہے اور اپنے استعمال میں لاتا ہے۔ جو فنکشنز ہم لکھتے ہیں اُنہیں یوزر ڈیفائنڈ فنکشنز کہا جاتا ہے۔ یوزر ڈیفائنڈ فنکشنز ایک قیمت دیتے ہیں اور عام طور پر ایسے اوپریشنز پر فارم کرتے ہیں جن کی ایک پروگرام میں بار بار ضرورت پڑتی ہے۔ بیسک میں یوزر ڈیفائنڈ فنکشنز کو پروسیجرز (Procedures) بھی کہا جاتا ہے اور یہ سب پروسیجرز کی طرح کام کرتے ہیں لیکن ان میں فرق یہ ہوتا ہے کہ یہ صرف ایک ویلیو دیتے ہیں۔ یوزر ان فنکشنز کو DEF FN کی سٹیٹمنٹ کے ذریعہ ڈیفائن (Define) کر سکتا ہے۔ یہ فنکشنز صرف ایک پروگرام کا حصہ ہوتے ہیں، BASIC لینگوئج کا حصہ نہیں ہوتے۔

ایک لائن کا فنکشن DEF FN سٹیٹمنٹ سے بنایا جاتا ہے جو کہ ایک ہی طرح کے کوڈ کو پروگرام میں مختلف جگہوں پر ایک سے زیادہ بار ایگزیکیوٹ (execute) کرتا ہے۔

فارمیٹ:

```
Line No DEF FN Name [arguments] expression
```

اس میں "name" سے مراد ایک ویری ایبل کا نام ہے۔ FN کے بعد آنے والا یہ نام فنکشن کے نام کا حصہ بن جاتا ہے۔

"arguments" وہ ویری ایبلز نام ہیں جن کو اُس وقت تبدیل کیا جاتا ہے جب فنکشن کو پروگرام میں کال کیا جائے۔ اس لسٹ میں تمام آئٹمز کو کومہ سے علیحدہ کیا جاتا ہے۔

"orpression" ایک ایسا ایکسپریشن ہے جو کہ فنکشن کا کام سرانجام دیتا ہے۔ اسے ایک سٹیٹمنٹ تک محدود کر دیا جاتا ہے۔

آرگیومنٹ کے ویری ایبلز، ایک سے ایک کی بنیاد پر، آرگیومنٹ ویری ایبلز یا ویلیوز کو ظاہر کرتے ہیں جو کہ فنکشن کال میں دینا پڑتے ہیں۔

یوزر ڈیفائنڈ فنکشن نومیرک یا سٹرنگ ہو سکتے ہیں۔ اگر فنکشن کے نام میں ٹائپ مخصوص کر دی جائے تو ایکسپریشن کی بھی وہی ٹائپ ہو جاتی ہے پیشتر اس کے کہ وہ کالنگ سٹیٹمنٹ کو واپس جائے۔ اگر فنکشن کے نام میں ایک ٹائپ مخصوص کر دی جائے اور آرگیومنٹ ٹائپ اس سے مختلف ہو تو "Type Mismatch" کی ایرر کا پیغام ملتا ہے۔

ایک یوزر ڈیفائنڈ فنکشن ایک پروگرام میں ایک سے زیادہ دفعہ ڈیفائن کیا جا سکتا ہے اور ایسا کرنے کے لیے DEF FN کی سٹیٹمنٹ کو دُھرانا پڑتا ہے۔

ایک DEF FN سٹیٹمنٹ لازمی طور پر پہلے ایگزیکیوٹ (Execute) ہوتی ہے اور بعد میں اس سے ڈیفائن کیے گئے فنکشن کو کال کیا جاتا ہے۔ اگر ایک فنکشن کو ڈیفائن کرنے سے پہلے کال کر لیا جائے تو "Undefined User Function" کی ایرر کا پیغام ملتا ہے۔

DEF FN کی ڈائریکٹ موڈ میں استعمال کی اجازت نہیں ہے۔ بار بار آنے والے فنکشنز کو DEF FN سٹیٹمنٹ سپورٹ نہیں کرتی۔ DEF FN سٹیٹمنٹ کا نام اگر سٹرنگ ویری ایبل پر مشتمل ہو تو اس کا جواب سٹرنگ کی شکل میں ہوتا ہے اور اگر ویری ایبل کا نام نومیرک ٹائپ کا ہو تو جواب نومیرک ویلیو میں ملتا ہے۔

مثال 1۔

```
10      REM
20      DEF FNX(Y) =(Y ^ 3+Y ^ 2)/Y
30      INPUT "Enter any two numbers", A,B
40      C = DEF FN X(A)+DEF FN X(B)
50      PRINT C
RUN
Enter any two numbers  4        5
50
```

مثال 2۔

```
10      CLS
20      PI = 3.1415
30      DEF FNR(X) = PI*X^ 2
40      INPUT "Radius="; RAD
50      PRINT "Circle Area is"; FNR(RAD)
60      END
RUN
Radius = 2
Circle Area is 12.556697
```

5.2 سب روٹینز (Subroutines)

ایک سب روٹین خودمختار سٹیٹمنٹس کے ایسے سیٹ پر مشتمل ہوتی ہے جسے کسی پروگرام میں کہیں سے بھی استعمال کیا جا سکتا ہے۔ سب روٹین خاص کام سرانجام دیتی ہے اور کنٹرول پروگرام کے اس حصے کو دوبارہ دے دیتی ہے جس نے سب روٹین کو کال کیا تھا۔ بسا اوقات سب روٹین کی سٹیٹمنٹس بنانا فنکشن بنانے کے مقابلے میں زیادہ آسان ہوتا ہے۔ سب روٹینز فنکشنز سے اس لحاظ سے ملتی جلتی ہیں کہ پروگرام کے مختلف حصوں میں ان کا ریفرنس دیا جا سکتا ہے۔ تاہم، فنکشن کے برعکس سب روٹین کو نام نہیں دیا جاتا اور اسے ایک سے زیادہ نومیرک اور سٹرنگ ویلیوز معلوم کرنے کے لیے استعمال کیا جا سکتا ہے۔ بیسک لینگوئج میں GOSUB-RETURN اور ON-GOSUB سٹیٹمنٹس سب روٹین کے لیے استعمال ہوتی ہیں۔

5.2.1 گوسب رٹرن سٹیٹمنٹ (GOSUB-Return Statement)

مقصد (Purpose): ایک سب روٹین میں جانے کے لیے اور اس سے واپس آنے کے لیے۔

وضاحت (Interpretation)

ایک سب روٹین کو ایک پروگرام میں حسب منشاء کال کیا جا سکتا ہے اور سب روٹین کو دوسری سب روٹین میں سے بھی کال کیا جا سکتا ہے۔ سب روٹینز کی ایسی نیسٹنگ (Nesting) موجود میموری (Memory) کے لحاظ سے محدود ہوتی ہے کیونکہ سسٹم کی میموری جتنی ہوا اتنی ہی نیسٹنگ ہوتی ہے۔

GW-BASIC میں ایک RETURN سٹیٹمنٹ کی وجہ سے اُس سٹیٹمنٹ تک واپس جایا جا سکتا ہے جو کہ موجودہ GOSUB سٹیٹمنٹ کے بعد آتی ہے۔ ایک سب روٹین میں ایک سے زیادہ RETURN سٹیٹمنٹس ہو سکتی ہیں۔ تاہم ضرورت کے مطابق RETURN سٹیٹمنٹ پروگرام میں مختلف مقامات پر آ سکتی ہے۔

سب روٹینز ایک پروگرام میں کسی جگہ پر بھی آ سکتی ہیں لیکن باقی پروگرام سے اس کی پہچان آسانی سے ہو جانی چاہیے۔

ایک سب روٹین سے پہلے STOP، END یا GOTO سٹیٹمنٹ آنی چاہیے تا کہ اس میں کوئی غلط اینٹر نہ ہو سکے اور پروگرام کنٹرول صحیح حصے کو منتقل ہو سکے۔

Syntax سینٹیکس

Line No. GOSUB *line number*

.

.

.

RETURN [line number]

لائن نمبر سب روٹین کی پہلی لائن نمبر ہے اور RETURN سٹیٹمنٹ میں اس کا استعمال اختیاری (Optional) ہے۔

مثال:

```
10    GOSUB 40
20    PRINT "BACK FROM SUBROUTINE"
30    END
```

```
40      PRINT "SUBROUTINE";
50      PRINT "IN";
60      PRINT "PROGRESS"
70      RETURN
RUN
SUBROUTINE IN PROGRESS
BACK FROM SUBROUTINE
```

اوپر والی مثال میں 30 نمبر لائن پر موجود END سٹیٹمنٹ کی وجہ سے سب روٹین دوبارہ ایگزیکیوٹ نہیں ہوتی۔

5.3 فائل ہینڈلنگ (File Handling)

ایک کمپیوٹر بہت زیادہ ڈیٹا (Data) پروسیس کرسکتا ہے۔ یہ جاننے کے لیے کہ فائلوں میں کس طرح لکھا جاسکتا ہے اور ان سے کیسے پڑھا جا سکتا ہے، ہمیں فائلوں کی بارے میں چند ضروری باتیں ضرور معلوم ہونی چاہئیں۔

کریکٹرز (Characters): کریکٹرز حروفِ تہجی، ہندسوں اور سپیشل کریکٹرز پر مشتمل ہوتے ہیں۔ انہیں کمپیوٹر میں ایک (1s) اور صفر (0s) کی مدد سے خاص ترتیب سے لکھا جاتا ہے۔

ڈیٹا فیلڈز (Data Fields): ڈیٹا فیلڈز متعلقہ حروف کے گروپ ہیں جو کہ خاص معلومات پر مشتمل ہوتے ہیں۔ مثال کے طور پر طالب علم کا نام، اس کا رول نمبر دو مختلف ڈیٹا فیلڈز ہیں۔

ریکارڈ (Record): متعلقہ فیلڈز کے گروپ کو ریکارڈ کہتے ہیں۔ مثال کے طور پر نام، باپ کا نام، رول نمبر، عمر، ایڈریس ایک طالب علم کا ریکارڈ ہے۔ اب تک ہم نے پروگرام کی مدد سے ڈیٹا حاصل کرنے کے لیے مختلف ٹیکنیکس (Technics) استعمال کی ہیں۔ مثال کے طور پر LET، INPUT اور READ/DATA سٹیٹمنٹ۔ ڈیٹا فائل میں ان پٹ اور آؤٹ پٹ سٹور کرنے کا ہمیشہ فائدہ ہوتا ہے۔

پروگرام فائلیں: پروگرام فائلیں پروگرام یا کمپیوٹر کے لیے ہدایات پر مشتمل ہوتی ہیں۔

ڈیٹا فائلز: ڈیٹا فائلز پروگراموں کے لیے درکار ڈیٹا یا انفرمیشن (Information) پر مشتمل ہوتی ہیں تاکہ یہ پروگرام صحیح طور پر کام کرسکیں۔ ڈیٹا فائلیں پروگرام چلنے کے دوران یا اس کی کمپائلیشن کے دوران ایک پروگرام فائل کے ساتھ منسلک (linked) ہوتی ہیں یا اس میں شامل (included) ہوتی ہیں۔

فائل تک ایکسیس کے طریقے (File Access Methods)

ایک فائل تک ایکسیس کے دو طریقے ہیں۔ جنہیں بالترتیب سیقوئینشل اور رینڈم (Random) کہا جاتا ہے۔

سیقوئینشل ایکسیس کا مطلب ہے کہ جو ڈیٹا مطلوبہ فائل میں محفوظ ہے اسے اس طرح سے ایکسیس کیا جائے گا جیسا کہ وہ ڈسک پر سٹور ہوا تھا۔ دوسرے لفظوں میں، اگر آپ ایک فائل کے پچیسویں (25^{th}) ریکارڈ تک رسائی چاہتے ہیں تو پہلے ریکارڈ نمبر ایک تا چوبیس تک رسائی ہوگی اور پھر پچیسویں ریکارڈ تک رسائی ہوگی۔

رینڈم ایکسیس کے طریقہ میں ایک پروگرامر کو ایک مخصوص ریکارڈ تک ڈائریکٹ رسائی (Direct access) حاصل ہو جاتی ہے۔ ظاہر ہے اس طرح سے ایک فائل سے ریکارڈ تلاش کرنا سیقوئینشل ایکسیس کے مقابلہ میں بہت آسان اور تیز تر ہو جاتا ہے۔

5.3.1 سیقوئینشل فائلز (Sequential Files)

یہ ایک اہم چیز ہے اور اسے اچھی طرح نوٹ کر لینا چاہیے کہ ایک فائل میں اگلے ریکارڈ کو پروسیس کرنے کے لیے ایک فائل پوزیشن پوائنٹر استعمال ہوتا ہے۔ سب سے پہلے، پوائنٹر ایک فائل کے شروع سے سٹارٹ (Start) ہوتا ہے اور فائل کے پہلے ریکارڈ کو پوائنٹ کرتا ہے اور پھر اس میں اضافہ ہوتا ہے اور یہ اگلے ریکارڈ تک پہنچا جاتا ہے۔ فائل پوائنٹر اس طرح فائل کے تمام ریکارڈز چیک کر لیتا ہے۔ سیقوئینشل ایکسیس کے طریقہ میں کام اس طرح ہوتا ہے۔ فرضی فائل پوزیشن پوائنٹر اصل میں ایک فائل کے ہر ریکارڈ میں سیقوئینشل حرکت کرتا ہے، یہاں تک کہ وہ مطلوبہ ریکارڈ تک پہنچ جاتا ہے۔

فائل اوپننگ (Opening File)

جب ایک سیقوئینشل فائل کو ایکسیس کیا جائے یا اُسے بنایا جائے تو پروگرامر کو سب سے پہلے فائل کھولنا پڑتی ہے اور اُس کے لیے OPEN سٹیٹمنٹ استعمال ہوتی ہے۔ OPEN سٹیٹمنٹ مندرجہ ذیل شکل میں دی جاتی ہے:

```
OPEN "FileName-ext" FOR mode AS #[buffer]
```

OPEN سٹیٹمنٹ فائل کا نام دیتی ہے، اس کے استعمال کا طریقہ (mode) بتاتی ہے اور فائل کا بفر (Buffer) نمبر بتاتی ہے۔ ڈیٹا فائلز کا نام رکھنے کے لیے وہی رولز استعمال ہوتے ہیں جو بیسک کی پروگرام فائلز کے نام رکھنے میں استعمال ہوتے ہیں، سوائے اس کے کہ ڈیٹا فائلز کو عام طور DAT کی ایکسٹینشن (Extension) دی جاتی ہے جبکہ ایک پروگرام فائل کی ایکسٹینشن BAS ہوتی ہے۔ اس کا مطلب یہ ہے کہ جو پروگرام ڈیٹا فائل استعمال کر رہا ہے اس کا نام ڈیٹا فائل کے نام جیسا ہو سکتا ہے۔ اُن کی ایکسٹینشنز کی مدد سے اُن میں فرق کیا جا سکتا ہے۔ پروگرام اور ڈیٹا فائل کا نام ایک جیسا اس وقت رکھتے ہیں جب دونوں ایک دوسرے سے مطابقت رکھتے ہوں۔

ایک فائل کو OUTPUT، INPUT یا APPEND کی طرح استعمال کیا جا سکتا ہے۔ یہ mode کے مختلف آپشنز ہیں۔ OUTPUT کا مطلب ہے کہ پروگرام آخر کار ڈیٹا کو فائل میں لکھے گا۔ INPUT کا مطلب ہے کہ جو ڈیٹا فائل میں موجود ہے وہ آخر کار کالنگ پروگرام میں لکھا جائے گا۔ APPEND کا مطلب ہے کہ ڈیٹا ایک موجودہ فائل کے آخر میں شامل ہو جائے گا۔

بفر پرائمری سٹوریج کا ایک ایسا حصہ ہے جہاں پر عارضی ڈیٹا جو کہ ایک فائل سے پڑھا جائے یا لکھا جائے سٹور ہوتا ہے۔ پروگرامر ایک فائل کا بفر نمبر مخصوص کر دیتا ہے جو کہ فائل کو پہچاننے اور ڈھونڈنے کے لیے ایک پروگرام میں استعمال ہوتا ہے۔ پروگرامر کو بفر نمبر لازمی طور پر معلوم ہونا چاہیے تا کہ وہ فائل میں موجود ڈیٹا پڑھ سکے یا لکھ سکے۔

اگر آپ ایسی فائل کھولنے کی کوشش کریں جو کہ پہلے سے موجود نہ ہو تو ایک نئی فائل بن جاتی ہے۔ اگر آپ ایک موجودہ فائل کو OUTPUT کے طور پر کھولیں گے تو اس میں موجود ڈیٹا ضائع ہو جائے گا اور اس کی جگہ نیا ڈیٹا لکھا جائے گا۔ مثال کے طور پر، مندرجہ ذیل OPEN سٹیٹمنٹ دیکھیں۔

```
OPEN "STUDATA.DAT" FOR OUTPUT AS #1
```

مندرجہ بالا OPEN سٹیٹمنٹ STUDATA.DAT فائل کو اوپن (Open) کرے گی تا کہ آؤٹ پُٹ کیا جا سکے اور نمبر 1 کو بفر نمبر کے طور پر استعمال کیا جائے گا۔ اگر STUDATA.DAT ڈسک پر پہلے سے موجود ہو تو اس کا ڈیٹا ضائع ہو جائے گا اور اس کی جگہ ایک خالی فائل لے لی گی۔ ایک دفعہ جب آپ نے کامیابی سے ایک فائل کھول لی تو آپ اس میں ڈیٹا لکھ سکتے ہیں، اس میں سے ڈیٹا پڑھ سکتے ہیں یا اس میں ریکارڈ کا اضافہ کیا جا سکتا ہے۔ ان تمام باتوں کا انحصار اس بات پر ہے کہ آپ نے فائل بناتے وقت کون سا موڈ مخصوص کیا۔

ایک فائل میں لکھنا (Writing to a File)

ایک فائل جو کہ OUTPUT کی طرح کھلی ہو اس میں لکھتے ہوئے ہم WRITE کی سٹیٹمنٹ استعمال کر سکتے ہیں۔ WRITE سٹیٹمنٹ بالکل PRINT سٹیٹمنٹ کی طرح کام کرتی ہے سوائے اس کے کہ آؤٹ پُٹ سکرین پر آنے کی بجائے، WRITE سٹیٹمنٹ اسے موجودہ اوپن فائل میں بھیج دیتی ہے۔ ایک خاص بات یہ ہے کہ آپ کو پہلے فائل کا بفر نمبر بھی بتانا پڑے گا جس میں ڈیٹا کو لکھنا ہے۔ مثال کے طور پر مندرجہ ذیل WRITE سٹیٹمنٹ کو دیکھیں ۔

```
Write1, name, address, phone
```

یہ سٹیٹمنٹ اُس ڈیٹا کو جو ڈیٹا name, address اور phone کے ویری ایبل میں سٹور ہے، اُسے بفر نمبر 1 دیتے ہوئے فائل میں لکھے گی۔ اصل میں #WRITE سٹیٹمنٹ ڈیٹا کو بفر نمبر 1 کے پرائمری سٹوریج لوکیشن پر منتقل کرتی ہے۔ پرائمری میموری سے سیکنڈری میموری میں ڈیٹا اُس وقت ٹرانسفر ہوتا ہے جب فائل کو "Close" کیا جاتا ہے۔ (اسے ہم کچھ دیر بعد پڑھیں گے۔)

ایک فائل سے پڑھنا (Reading from a File)

ایک فائل جو کہ INPUT کی طرح کھلی ہو اُس سے ڈیٹا پڑھنے کرنے کے لیے INPUT سٹیٹمنٹ استعمال ہوتی ہے۔ INPUT بالکل INPUT سٹیٹمنٹ کی طرح ہے، سوائے اس کے کہ کی بورڈ سے دیا گیا یوزر کا ڈیٹا ریڈ (Read) کرنے کی بجائے، INPUT سٹیٹمنٹ ایک فائل سے ڈیٹا ریڈ کرتی ہے جو کہ ڈِسک پر موجود ہوتی ہے۔ مثال کے طور پر مندرجہ ذیل کو دیکھیں:

```
INPUT 1, stuName, gpa, grade
```

یہ سٹیٹمنٹ بفر نمبر 1 میں موجود فائل سے ویلیوز لیتی ہے اور سٹیٹمنٹ میں دیے گئے ویری ایبلز کو ایک، ایک کر کے قیمت دیتی ہے۔ یہ اہم بات نوٹ کرنے کی ہے کہ INPUT سٹیٹمنٹ اُس ریکارڈ سے جو کہ فائل پوزیشن پوائنٹر کی مدد سے اِس وقت پوائنٹ کیا گیا ہو، ڈیٹا ریڈ کرتی ہے۔ جب INPUT سٹیٹمنٹ ایگزیکیوٹ ہوتی ہے تو فائل پوزیشن پوائنٹر کی ویلیو میں ایک کا اضافہ ہو جاتا ہے اور یہ فائل میں موجود اگلے ریکارڈ پر پہنچ جاتا ہے۔ اگر ہمیں فائل میں موجود مزید ڈیٹا کو ریڈ کرنا ہو تو یہ پروسیس پھر دہرایا جاتا ہے۔

یہ جاننا بھی ضروری ہے کہ ہر فائل کے آخر میں ایک End-of-file کریکٹر ہوتا ہے۔ اس کی مدد سے جو پروگرام فائل کو پڑھ رہا ہو اسے علم ہو جاتا ہے کہ اب فائل میں کوئی اور ڈیٹا نہیں ہے۔ جب ہم فائل سے ڈیٹا ریڈ کر رہے ہوتے ہیں تو End-of-file کریکٹر سے ہمیں یہ جاننے میں مدد ملتی ہے کہ تمام ڈیٹا پڑھ لیا گیا ہے۔ EOF() فنکشن پڑھنے کے عمل کے دوران فائل کا اختتام (end) جاننے میں مدد دیتا ہے۔ اگر فائل میں اور ریکارڈ موجود ہوئے تو EOF() فنکشن غلط (false) کی ویلیو یعنی صفر (0) دے گا۔ اگر فائل میں ریکارڈ

موجود نہ ہوں اور اس کا آخری حصہ آ جائے تو یہ درست (true) کی ویلیو (1) دے گا۔ اس لیے EOF() فنکشن دراصل ایک بولین ایکسپریشن کی طرح معلوم کیا جاتا ہے۔ EOF() فنکشن کے ساتھ بھیجا گیا آرگیومینٹ زیرِ استعمال فائل کا بفر نمبر ہو گا۔ چونکہ EOF() فنکشن ایک بولین ایکسپریشن ہے جو true یا false کی شکل میں جواب دیتا ہے، یہ ایک فائل سے ڈیٹا پڑھنے کے دوران لُوپ کو کنٹرول کرنے کا ایک عمدہ طریقہ ہے۔ مثال کے طور پر، دیکھیں:

```
10      DO WHILE NOT EOF(1)
20      INPUT1, stuName, gpa, grade
30      PRINT "STUDENT:"; stuName; SPC(5); gpa; SPC(5); grade
40      WHILE
```

مندرجہ بالا کوڈ تمام ڈیٹا اور ریکارڈز جو کہ فائل میں موجود ہوں اور اُن کا بفر نمبر 1 ہو، کو ریڈ کرتا ہے جب تک کہ end-of-file کا نشان نہ آ جائے۔ اگر ہم چاہتے ہیں کہ ایک ڈیٹا جو کہ ایک خاص ریکارڈ اور ایک فائل میں موجود ہو اُسے سیقوئینشلی (Sequentially) ایکسیس کرنے کے لیے ہمیں جو ریکارڈ چاہیے اس تک سارے ریکارڈز کو ریڈ کرنا پڑے گا (نہ کہ پرنٹ)۔ اس چیز کو ہم ایک "dummy" ویری ایبل بنا کر جان سکتے ہیں کہ مطلوبہ ریکارڈ تک پہنچنے کی لیے کتنے ریکارڈ پڑھے گئے ہیں۔ اس میں کافی وقت ضائع ہوتا ہے۔ اس مسئلہ کا اس سے بہتر حل یہ ہے کہ رینڈم ایکسیس طریقہ استعمال کیا جائے۔ ہم رینڈم ایکسیس کو آگے چل کر پڑھیں گے۔

ایک فائل کو کلوز کرنا (Closing a File)

فائل استعمال کرنے کے بعد اسے لازمی طور پر بند کرنا چاہیے۔ جب ہم ڈسک پر موجود فائل میں ڈیٹا لکھ رہے ہوں تو Close سٹیٹمنٹ کی یہ ذمہ داری ہوتی ہے کہ پرائمری میموری میں موجود ڈیٹا کو سیکنڈری میموری میں لے جائے۔ اس لیے کسی بھی اوپن فائل کو کلوز کرنا ضروری ہوتا ہے۔ Close سٹیٹمنٹ کا استعمال مندرجہ ذیل طریقہ سے کیا جاتا ہے۔

```
CLOSE #buffer
```

جب CLOSE کی سٹیٹمنٹ آتی ہے تو ڈیٹا فائل پر لکھ دیا جاتا ہے۔ یہ متعلقہ بفر میں موجود تمام عارضی ڈیٹا کو فائل میں لکھ دیتی ہے۔ اگر ہم اپنی پچھلی STUDATA.DAT فائل کو کلوز کرنا چاہیں تو ہم اسے درج ذیل طریقے سے استعمال کرتے ہیں۔

```
CLOSE #1
```

آپ ایک سے زیادہ اوپن فائلز ایک ہی CLOSE سٹیٹمنٹ کے ذریعہ بند کر سکتے ہیں۔ جیسا کہ

```
CLOSE #1, #2, #3, ..... #n
```

آپ تمام موجودہ اوپن فائلز کو ایک سنگل CLOSE سٹیٹمنٹ سے بھی کلوز کر سکتے ہیں۔ جیسا کہ

```
CLOSE
```

اب آپ ایک مکمل پروگرام ملاحظہ کریں جو کہ دکھاتا ہے کہ ڈیٹا کو کیسے لکھا اور پڑھا جا سکتا ہے۔ اس میں ہم سیقوئینشل فائل کا استعمال دیکھیں گے اور اس کے بعد رینڈم فائل کو پڑھیں گے۔

```
REM TOPIC CONCENTRATION: SEQUENTIAL FILES
REM This program will create a sequential file named INFO.DAT
REM The file will then be used to store input data by user
REM and then to retrieve the data shared in file
OPEN "INFO.DAT" FOR OUTPUT AS #1
CLS
REM Get input from user
INPUT "Do you wish to enter student information (Y/N)"; ANSWER$
IF ANSWER$ = "n" OR ANSWER$ = "N" THEN NODATA = 1 ELSE  NODATA
= 2
WHILE ANSWER$ = "Y" OR ANSWER$ = "y"
  INPUT "Enter student name:",NAME$
  INPUT "Enter student id: ", ID
  INPUT "Enter student GPA: ", GPA
REM Write current input to temporary buffer storage 1
  WRITE #1, NAME$, ID, GPA
   INPUT "Do you wish to enter new student information (Y/N)";
  ANSWER$
WEND
REM Transfer data from temporary buffer 1 to actual INFO.DAT
CLOSE #1
OPEN "INFO.DAT" FOR INPUT AS #2
IF NODATA<> 1 THEN
    PRINT
    PRINT "STUDENT LISTING"
     PRINT "---------------------------------------- "
END IF
WHILE NOT EOF(2)
    INPUT #2, NAME$, ID, GPA
```

```
    PRINT
    PRINT "STUDENT NAME: "; NAME$
    PRINT "  STUDENT ID:"; ID
    PRINT " STUDENT GPA:"; GPA
WEND
CLOSE #2
END
```

5.3.2 رینڈم فائلز (Random Fales)

آپ جس پرابلم پر کام کر رہے ہیں اگر اس کا حل اس طرح سے ہے کہ آپ کو ایک فائل میں ایک خاص ریکارڈ تک ڈائریکٹ پہنچنا ہو تو اِس کا تیز ترین الگورتھم یہ ہوگا کہ آپ رینڈم فائل ایکسیس طریقہ استعمال کریں۔ جیسا کہ پہلے بتایا جا چکا ہے کہ سیقوئینشل فائلز کے برعکس رینڈم فائلز ایک فائل میں تمام پچھلے ریکارڈز کو ایکسیس کیے بغیر مطلوبہ ریکارڈ کو ڈائریکٹ ایکسیس کر سکتی ہیں۔ ایک فائل کو مخصوص کرنا کہ اسے رینڈم سمجھا جائے یہ اُسی طرح سے ہے جیسا کہ ایک سیقوئینشل فائل میں مخصوص کیا جائے کہ مختلف موڈ اوپریشنز کیسے ہوتے ہیں۔ جیسا کہ دیگر اقسام کی تمام فائلوں میں ہوتا ہے، پروگرام کو فائل استعمال کرنے سے پہلے اسے اوپن کرنا چاہیے۔ فائل کھولنے کا طریقہ درج ذیل ہے۔

OPEN "File Name.ext" FOR RANDOM AS # [buffer] LEN = LEN (Record Variable)

نوٹ کریں کہ ہم نے RANDOM فائل اوپن کی ہے اور معمول کے مطابق بفر فائل نمبر استعمال کیا گیا ہے۔ سب سے بڑا فرق LEN = LEN() فنکشن کا ہے۔

آرگیومنٹ (Argument) ریکارڈ ویری ایبل جو کہ LEN() فنکشن کو بھیجا جاتا ہے ریکارڈ ٹائپ ویری ایبل ہے۔ LEN() فنکشن ریکارڈ ویری ایبل کا سائز (Size) واپس دیتا ہے تا کہ فائل میں ہر ریکارڈ کو سٹور کرنے کی جگہ مخصوص کی جا سکے۔

رائٹنگ یا سٹورنگ ریکارڈز (Writing or Storing Records)

جب ہم رینڈم فائلز استعمال کر رہے ہوں، تو ہم فائل کے ہر فیلڈ میں ایک، ایک کر کے ڈیٹا کو نہیں بھیج سکتے۔ اس کی بجائے، رینڈم فائل کی مدد سے ہم پورے ریکارڈ کے کونٹینٹس (Contents) کو ایک ساتھ، ایک ہی سٹیٹمنٹ میں فائل میں بھیج دیتے ہیں۔ ایسا ہم PUT سٹیٹمنٹ کے استعمال سے کرتے ہیں۔ PUT سٹیٹمنٹ یہ بھی بتاتی ہے کہ ریکارڈ فائل میں کس جگہ پر بھیجا جا رہا ہے۔ PUT سٹیٹمنٹ کی فارم (Form) درج ذیل ہے۔

PUT #[buffer], recordNumber, recordVariable

مندرجہ بالا سٹیٹمنٹ میں بفر فائل نمبر لازمی طور پر بتانا پڑتا ہے، ریکارڈ کا نمبر یا ریکارڈ کی لوکیشن جہاں پر فائل میں اسے بھیجنا ہوتا ہے وہ اس کے بعد آتی ہے اور آخر میں ریکارڈ ویری ایبل بتایا جاتا ہے تا کہ پروگرام کو پتہ چل سکے کہ کون سا ریکارڈ فائل میں بھیجنا ہے۔

PUT سٹیٹمنٹ کے بالکل اُلٹ عمل ڈیٹا فائل میں سے ڈیٹا حاصل کرنا ہے۔اس کام کے لیے بیسک میں ایک بلٹ-ان فنکشن ہے۔اسے GET کہا جاتا ہے اور اس کی فارم PUT سٹیٹمنٹ کی طرح کی ہوتی ہے۔GET کی فارم مندرجہ ذیل ہے۔

GET #[buffer], recordNumber, recordVariable

جب آپ GET سٹیٹمنٹ استعمال کر رہے ہوں، تو کسی بھی ریکارڈ کو، ریکارڈ نمبر بتاتے ہوئے یا فائل نمبر کی لوکیشن بتاتے ہوئے، براہ راست ایکسیس کیا جا سکتا ہے۔اس وجہ سے رینڈم فائلز، سیقوئینشل فائلز کے مقابلہ میں بہت زیادہ بہتر شمار ہوتی ہیں۔ہم رینڈم فائلز بنا سکتے ہیں لیکن فی الوقت یہ نصاب میں شامل نہیں ہے۔

مشق

-1 خالی جگہ پر کریں۔

(i) ایک سب روٹین بذاتِ خود ایک ----------پروگرام ہے۔

(ii) ایک فنکشن ----------ویلیو کیلکولیٹ کرنے اور دینے کے لیے استعمال ہوتا ہے۔

(iii) GOSUB اور GOTO سٹیٹمنٹس ----------ہیں۔

(iv) GOSUB سٹیٹمنٹ ----------میں جانے کے لیے استعمال ہوتی ہے۔

(v) یوزر ڈیفائنڈ فنکشن نومیرک یا ----------ہو سکتا ہے۔

(vi) INT (7.6) برابر ہے ----------کے۔

(vii) یوزر ڈیفائنڈ فنکشن کی ورڈ ----------سے شروع کیا جاتا ہے۔

(viii) GOSUB سٹیٹمنٹ ---------- کی سٹیٹمنٹ کے ساتھ ختم ہونی چاہیے۔

(ix) RND فنکشن کا ذکر ----------نمبر جنریٹر کے طور پر کیا جاتا ہے۔

(x) ----------فیلڈز کا مجموعہ ہے جسے فائل میں کسی چیز کے بارے میں معلومات دینے کے لیے استعمال کیا جاتا ہے۔

-2 درست جواب کا انتخاب۔

(i) LEFT$ ("Pakistan",3) برابر ہے ----------کے۔

'Pak' (a) 'PAK' (b) ' Pa' (c) 'kis' (d)

(ii) فنکشن INT(-5.7) کا جواب ہے:

-5 (a) -4 (b) -6 (c) 5 (d)

(iii) RIGHT$(x$,n)

(a) سٹرنگ x$ کے دائیں طرف 'n' خالی جگہیں چھوڑ دے گا

(b) سٹرنگ x$ کے بائیں طرف 'n' خالی جگہیں چھوڑ دے گا

(c) سٹرنگ x$ کے دائیں طرف 'n' خالی جگہیں منتخب کرے گا

(d) سٹرنگ x$ کے بائیں طرف 'n' خالی جگہیں منتخب کرے گا

(iv) ---------- فنکشن ASCII کوڈز کو متبادل کریکٹر میں تبدیل کرنے کے لیے استعمال ہوتا ہے۔

(a) CHAR$() (b) CHR() (c) CHAR. (d) CHR$()

(v) ہدایات جو کہ ایک بار مین پروگرام میں لکھی جاتی ہیں یا الگ سے لکھی جاتی ہیں اور انہیں مین پروگرام سے ایک سے زیادہ بار کال کیا جاسکتا ہے، کہلاتی ہیں۔

(a) کنٹرول سٹیٹمنٹ (b) لُوپ (c) سب پروگرام (d) ان میں سے کوئی نہیں

(vi) TAN(x) برابر ہے۔

(a) SIN(x)/COT(x) (b) COT(x)

(c) COS(x) / SIN(x) (d) SIN(x)/COS(x)

(vii) SIN(-4) کا آؤٹ پٹ ہے۔

(a) '_' sign (b) '+' sign (c) 0 (d) ان میں سے کوئی نہیں

(viii) ایک فائل اپنے استعمال سے پہلے ان میں سے کسی ایک سٹیٹمنٹ سے ایکٹیویٹ کی جاتی ہے:

(a) WRITE (b) READ (c) PRINT (d) ان میں سے کوئی نہیں

(ix) ایک فائل درج ذیل طریقوں سے بہتر طور پر ہینڈل کی جاسکتی ہے:

(a) ایک طریقہ (b) دو طریقے (c) تین طریقے (d) چار طریقے

(x) فائل سے معلومات پڑھنے کے لیے اسے کھولنا چاہیے:

(a) ان پٹ (b) آؤٹ پٹ (c) a اور b دونوں (d) ان میں سے کوئی نہیں

3- صحیح یا غلط کی نشاندہی کریں۔

(i) فائلوں کو بہتر طور پر ہینڈل کرنے کے تین طریقے ہیں۔

(ii) سٹیٹمنٹ END مین پروگرام کے آخر پر دی جانی چاہیے۔

(iii) FIX فنکشن کسری حصہ ختم کر کے صحیح عدد دیتا ہے۔

(iv) سب روٹینز ڈیزائن کرنا، ان کی ایررز درست کرنا اور ان میں تبدیلی کرنا آسان کام نہیں ہے۔

(v) ایک فنکشن دو سٹیٹمنٹس GOSUB اور RETURN سے منسلک ہوتا ہے۔

(vi) فائل ریکارڈز کا مترتب مجموعہ ہوتی ہے۔

(vii) OPEN سٹیٹمنٹ میں فائل کا حوالہ ضرور ہونا چاہیے۔

(viii) بلٹ-اِن فنکشن کو لائبریری فنکشن کے طور پر بھی ڈیفائن کیا جا سکتا ہے۔

(ix) آؤٹ پٹ کے لیے کھولی گئی فائل کو اس سے ڈیٹا پڑھنے کے لیے استعمال کیا جاتا ہے۔

(x) سب روٹینز استعمال کرنے کے لیے استعمال ہونے والے کی ورڈز GOSUB اور RETURN ہیں۔

4۔ یوزر۔ڈیفائنڈ فنکشن اور بلٹ-اِن فنکشن میں کیا فرق ہے؟

5۔ سب روٹین اور فنکشن میں فرق بتائیں۔

6۔ GOSUB-RETURN سٹیٹمنٹ کا استعمال بتائیں۔

7۔ سیقوئینشل فائلز اور رینڈم فائلز میں کیا فرق ہے؟

8۔ ایک سیقوئینشل فائل کے اوپننگ، کلوزنگ، ریڈنگ اور رائٹنگ کے مختلف طریقے بیان کریں۔

9۔ ڈیٹا فائل اور پروگرام فائل میں فرق بیان کریں۔

10۔ مندرجہ ذیل فنکشنز کا مقصد لکھیں:

ABS() (i) INT() (ii) SQR() (iii) SIN() (iv) COS() (v)

11۔ کسی شخص کا پورا نام لینے اور اُس کے نام کے پہلے حصے (first name) میں کریکٹرز کی تعداد بتانے کے لیے پروگرام تحریر کریں۔

12۔ 1 تا 255 تک ASCII کریکٹرز پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

13۔ DEF FN فنکشن کی مدد سے سیلشیس سکیل سے فارن ہائیٹ سکیل میں تبدیلی کے لیے پروگرام تحریر کریں۔

14۔ یوزر ڈیفائنڈ فنکشن کے استعمال سے درج ذیل فارمولے کے مطابق کیلکولیٹ کرنے اور رزلٹ پرنٹ کرنے کے لیے پروگرام تحریر کریں۔

Combination = n!/k!(n-k)!

15۔ سیقوئینشل ایکسیس فائلز کو استعمال کرتے ہوئے ٹیلی فون ڈائریکٹری کو امپلیمنٹ کرنے کا پروگرام لکھیں۔ آپ کا پروگرام اس قابل ہو کہ آپ کے دوستوں کے نام، ٹیلی فون نمبر اور پتے ایک سیقوئینشل فائل میں لکھے۔

جوابات

1- (i) چھوٹا (ii) سنگل

(iii) مختلف (iv) سب روٹین

(v) سٹرنگ (vi) 7

(vii) DEF FM (viii) RETURN

(ix) رینڈم (x) ریکارڈ

2- (i) c (ii) c

(iii) c (iv) d

(v) c (vi) d

(vii) d (viii) d

(ix) b (x) a

3- (i) غلط (ii) صحیح

(iii) غلط (iv) غلط

(v) غلط (vi) صحیح

(vii) صحیح (viii) صحیح

(ix) غلط (x) صحیح

باب6

بیسک میں گرافِکس

(GRAPIHCS IN BASIC)

6.1 تعارف (Introduction)

معلومات کو تصویری شکل میں ڈیزائن کرنے اور پیش کرنے کے فن کو گرافکس کہتے ہیں۔ یہ سہولت بیسک لینگوئج کے تقریباً تمام ورژنز میں میسر ہے۔ اس کی مدد سے سکرین پر معلومات ڈسپلے (Display) کی جاتی ہیں۔ آج کل کمپیوٹر پروگرامنگ میں گرافکس کا استعمال بہت زیادہ ہے۔ اس کے استعمال کا دارومدار ہارڈ ویئر جیسا کہ ان پٹ، آؤٹ پٹ اور گرافک کارڈ (کلر گرافک ایڈاپٹر، ویڈیو گرافک ارے) پر ہے۔

آیئے ہم شروع سے آغاز کریں۔ آپ کی سکرین سینکڑوں پکسلز (Pixels) پر مشتمل ہے۔ افقی (Horizontal) اور عمودی (Vertical) پکسلز کی تعداد سے مونیٹر کی ریزولوشن (Resolution) کا علم ہوتا ہے۔ GW-BASIC میں یہ کسی ایک گرافک موڈ میں ہوسکتی ہے جس سے گرافکس کی ریزولوشن (پکسلز)، ٹیکسٹ ریزولوشن (کریکٹرز)، رنگوں کی تعداد اور ویڈیو صفحات کی تعداد کا علم ہوتا ہے۔ سکرین گرافک موڈ تیرہ (13) ہیں اور ہر ایک کا مقصد جدا ہے۔ سکرین پر ڈرائنگ کرنے کے کئی طریقے ہیں۔ ڈرائنگ کے لیے سکرین کا ایریا مقرر کرنے کے لیے کوآرڈینیٹس (Coordinates) استعمال ہوتے ہیں۔ ایک کوآرڈی نیٹ سے مراد ایک خاص پکسل (تصویری ایلیمنٹ: سکرین پر ایک نقطہ کو پکسل کہتے ہیں) ہے۔ آپ کی کمپیوٹر سکرین تقریباً 1 ملین چھوٹے چھوٹے رنگین مربعوں میں بنی ہوتی ہے۔ ہر چھوٹا مربع (Square) ایک پکسل کہلاتا ہے۔ آپ کسی بھی کوآرڈینیٹ کی پوزیشن کا تعین کر سکتے ہیں۔ اس کے لیے آپ کو نیچے کے پکسلز کی تعداد اور دائیں طرف کے پکسلز کی تعداد گننا پڑے گی۔ بیسک میں ڈیٹا ڈسپلے کرنے کے تین موڈز ہیں۔

☆ ٹیکسٹ موڈ ☆ میڈیم۔ ریزولوشن گرافک موڈ ☆ ہائی۔ ریزولوشن گرافک موڈ

ٹیکسٹ موڈ (Text Mode)

یہ موڈ ٹیکسٹ ٹائپ ڈیٹا کے لیے استعمال ہوتا ہے۔ ٹیکسٹ بیسڈ گرافک میں سکرین پر ٹیکسٹ اور لائنیں کھینچی جاسکتی ہیں۔ ڈسپلے سکرین کو 80×25 پکسلز کے میٹرکس میں تقسیم کیا جاتا ہے۔ اس موڈ میں 2 رنگوں کے کسی بھی کلر پیلٹس (Color palattes) کے 16 رنگ پرنٹ کیے جاسکتے ہیں (ٹیبل 6.1)۔ کالمز کے نمبر 0 تا 39 یا 79 تک ہوتے ہیں جبکہ قطاروں کے نمبر 0 تا 24 تک ہوتے ہیں۔

میڈیم۔ ریزولوشن گرافک موڈ (Medium - Resolution Mode)

میڈیم۔ ریزولوشن گرافک موڈ گرافک بنانے کے لیے استعمال ہوتا ہے۔ ڈسپلے سکرین کو 320×200 پکسلز کے میٹرکس میں تقسیم کیا جاتا ہے۔ اس طرح ہر ایک پکسل کی پوزیشن کا تعین سکرین کے ایکس اور وائی کوآرڈینیٹس سے کیا جاتا ہے۔ یہ گرافیکل موڈ 4 کلرز کے ساتھ کام کرتا ہے۔ چار مختلف رنگ 0، 1، 2 اور 3 ہیں۔ سولہ رنگوں میں سے ایک رنگ کو بیک گراؤنڈ (Background) کے لیے اور ایک رنگ فرنٹ (Front) کے لیے منتخب کیا جاتا ہے۔ فرنٹ اور بیک گراؤنڈ پیلٹس کی فہرست ٹیبل 6.1 میں دی گئی ہے۔

ہائی-ریزولوشن گرافک موڈ (High - Resolution Graphic Mode)

ہائی-ریزولوشن گرافک موڈ 200 ×640 پکسلز کے میٹرکس میں گرافکس بنانے کے لیے استعمال ہوتا ہے۔ ہم ٹیکسٹ کریکٹرز کو 25 لائنوں میں ظاہر کر سکتے ہیں جبکہ ہر ایک قطار میں 80 کریکٹرز آ سکتے ہیں۔

6.2.1 سکرین سٹیٹمنٹ (SCREEN Statement)

SCREEN سٹیٹمنٹ عام طور سے ایک سکرین موڈ کو سلیکٹ کرنے کے لیے استعمال ہوتی ہے جو کہ ایک مخصوص ڈسپلے ہارڈ ویئر کو کونفیگر (Configure) کرنے کے لیے مناسب ہوتا ہے۔ مثال کے طور پر IBM مونوکروم (Monochrome) ڈسپلے اور پرنٹر ایڈاپٹر (Adopter) (MDPA) کے ساتھ موڈ صفر (0) استعمال ہوتا ہے۔ اس کونفگریشن (Configuration) کے لیے لکھے گئے پروگرام صرف ٹیکسٹ موڈ میں ہونے چاہئیں۔

سینٹیکس Syntax

SCREEN [mode][,[colorswitch]]

جبکہ سکرین موڈ سے مراد 0، 1، 2، 7، 8، 9، 10 تک نومیرک ویلیو ہے۔ اسے ہر قسم کے کمپیوٹر یا مونیٹر پر نہیں چلایا جا سکتا کیونکہ اس کا دارمدار ڈسپلے کارڈ اور مونیٹر کی ٹائپ پر ہوتا ہے۔ سکرین کے مختلف موڈز نیچے دیے گئے ہیں۔

Screen modes:- 0,1,2,7,8,9,10

بیسک لینگوئج میں موجودہ سکرین موڈ 0 ہوتا ہے۔ یہ صرف ٹیکسٹ بیسڈ ہے اور اسے سکرین پر دیکھا جا سکتا ہے۔ سکرین موڈ 1 میڈیم-ریزولوشن گرافک موڈ کو ایکٹیویٹ (Activate) کرتا ہے۔ اس میں 200 ×320 پکسلز کی ریزولوشن والے گرافکس پرنٹ کیے جا سکتے ہیں۔ سکرین موڈ 2 ہائی ریزولوشن گرافک موڈ کو ایکٹیویٹ کرتا ہے۔ اس میں 200 ×640 پکسلز کے گرافکس حاصل کیے جا سکتے ہیں۔ اس کے لیے اچھی کوالٹی کے مونیٹرز مثلاً سی جی اے (کلر گرافکس اڈاپٹر)، ای جی اے (انہانسڈ گرافکس اڈاپٹر)، وی جی اے وغیرہ کی ضرورت ہوتی ہے۔

سکرین موڈز 7,8,9 اور 10 میڈیم-ریزولوشن اور ہائی-ریزولوشن کے گرافکس بنانے کے لیے استعمال ہوتے ہیں۔ یہ تمام گرافکس کے لیے استعمال ہوتے ہیں اور ان سے اچھی کوالٹی کا رزلٹ حاصل کیا جا سکتا ہے۔ کچھ اہم سکرین موڈز کا تعارف درج ذیل ہے۔

سکرین موڈ	وضاحت
0	ٹیکسٹ موڈ، اس میں گرافکس استعمال نہیں ہو سکتے
1	200 × 320 ، صرف 4 کلرز
2	200 × 640 ، صرف 2 کلرز (سیاہ اور سفید)
7	200 × 320 ، کل 16 کلرز اور پیجز (Pages) کو سپورٹ کرتا ہے۔ یہ سکرین موڈ بہت مفید ہے۔
8	200 × 640 ، سولہ (16) کلرز اور پیجز کو سپورٹ نہیں کرتا

9	350 × 640 ، کل 16 کلرز اور پیجز کو سپورٹ کرتا ہے
10	350 × 640 ، صرف 2 کلرز (سیاہ اور سفید)
11	480 × 640 ، صرف 2 کلرز
12	480 × 640 ، صرف 16 کلرز (بہت مفید)

مختلف سکرین موڈز اور ڈسپلے ہارڈ ویئر کو نفگریشنز کے لیے مختلف ایٹریبیوٹس (Attributes) اور کلر سیٹنگز ہیں۔ (ایٹریبیوٹ اور کلر نمبر پر بحث کے لیے PALETTE کی سٹیٹمنٹ دیکھیں) زیادہ تر سکرین موڈز کے موجودہ ایٹریبیوٹس اور سکرین موڈز درج ذیل ہیں:

Attributes for Mode				Color Display	
1,9	2	0,7,8,9	نمبر	Color	رنگ
0	0	0	0	Black	کالا
		1	1	Blue	نیلا
		2	2	Green	سبز
		3	3	Cyan	گہرا سبز نیلا
		4	4	Red	سرخ
		5	5	Magenta	گہرا گلابی
		6	6	Brown	بھورا
		7	7	White	سفید
		8	8	Gray	سرمئی
		9	9	Light Blue	ہلکا نیلا
		10	10	Light Green	ہلکا سبز
1		11	11	Light Cyan	ہلکا سبز نیلا
		12	12	Light Red	ہلکا سرخ
2		13	13	Light Magenta	ہلکا گلابی
		14	14	Yellow	پیلا، زرد
3	1	15	15	High-intensity White	نہایت سفید

ٹیبل 6.1: سکرین 10 کے علاوہ سکرین کے لیے کلر ایٹریبیوٹ

6.2.2 COLOR سٹیٹمنٹ (COLOR Statement)

سکرین موڈز کے بعد گرافکس میں استعمال ہونے والی اگلی چیز COLOR سٹیٹمنٹ ہے۔ COLOR سٹیٹمنٹ کا مقصد ڈسپلے کلر سلیکٹ کرنا ہے۔

سینٹیکس Syntax

COLOR [foreground][,[background][,border]]

COLOR [background][,[palette]]

COLOR [foreground][,[background]]

عام طور پر، COLOR سٹیٹمنٹ سامنے کے اور بیک گراؤنڈ کے کلرز سلیکٹ کرنے کے لیے استعمال کی جاتی ہے۔ SCREEN 0 میں بارڈر کا کلر بھی سلیکٹ کیا جا سکتا ہے۔ SCREEN 1 میں سامنے کا کوئی کلر بھی سلیکٹ کیا جا سکتا ہے مگر چار کلر کے دو پیلٹس میں سے کسی ایک کو گرافک سٹیٹمنٹس کے ساتھ استعمال کے لیے سلیکٹ کیا جا سکتا ہے۔ مختلف سکرین موڈز، سینٹیکسز اور ان کے اثرات کا ذکر درج ذیل ہے:

موڈ	اثر
SCREEN 0	یہ موڈ موجودہ ٹیکسٹ کے فورگراؤنڈ (Foreground) اور بیک گراؤنڈ کلرز اور سکرین بارڈر تبدیل کرتا ہے۔ فورگراؤنڈ کلر 31-0 تک کی رینج میں سے صحیح عدد پر مشتمل ایکسپریشن ہونا چاہیے۔ یہ ٹیکسٹ موڈ میں فورگراؤنڈ کلر مخصوص کرنے کے لیے استعمال ہوتا ہے جو کہ ٹیکسٹ کا موجودہ کلر ہے۔ صحیح اعداد 15-0 کی رینج میں سے سولہ کلرز منتخب کیے جا سکتے ہیں۔ کلر نمبر میں 16 کے اضافے سے بلنکنگ (Blinking) کلر بھی سلیکٹ کیا جا سکتا ہے؛ مثال کے طور پر، کلر 7 کا بلنکنگ کلر 16+7 یعنی 23 ہوگا۔ اس لیے، فورگراؤنڈ کی لیگل رینج 31-0 ہے۔
	بیک گراؤنڈ کلر 7-0 تک کی رینج میں سے صحیح عدد ہونا چاہیے اور یہ ہر ٹیکسٹ کریکٹر کا بیک گراؤنڈ کلر ہے۔ یہاں بلنکنگ کلرز نہیں دیے جا سکتے۔
	بارڈر کلر 15-0 کی رینج میں سے کسی صحیح عدد پر مشتمل ہوتا ہے اور یہ وہ کلر ہے جو سکرین بارڈر میں استعمال ہوتا ہے۔ یہاں بلنکنگ کلرز نہیں دیے جا سکتے۔
	اگر COLOR سٹیٹمنٹ کے ساتھ کوئی آرگیومنٹ نہ دیا جائے تو بیک گراؤنڈ اور بارڈر کا ڈیفالٹ (Default) کلر سیاہ (COLOR 0) ہوتا ہے اور فورگراؤنڈ کلر کا ذکر SCREEN سٹیٹمنٹ کے ریفرنس (Reference) پیجز میں بتائے گئے کلر کے مطابق ہوتا ہے۔
SCREEN 1	موڈ 1 میں سٹیٹمنٹ کا سینٹیکس نہایت مختلف ہوتا ہے جس میں پیلٹ کا آرگیومنٹ بھی شامل ہوتا ہے جو کہ کسی طاق یا جفت صحیح ایکسپریشن پر مشتمل ہوتا ہے۔ یہ آرگیومنٹ خاص تعداد میں کلرز ڈسپلے کرنے کے لیے ڈسپلے کلر کے سیٹ کو متعین کرتا ہے۔
	ہارڈ ویئر کونفگریشنز کے لیے پیلٹ پیرامیٹر (Parameter) کی موجودہ کلر سیٹنگ (Color setting) درج ذیل ہے:

COLOR ,0 'Same as the next three PALETTE statements '1 = green, 2 = red, 3 = yellow COLOR ,1 'Same as the next three PALETTE statements '1 = cyan, 2 = magenta, 3 = white	
اس موڈ میں کچھ بھی نہیں ہوتا۔ اگر اس موڈ میں COLOR سٹیٹمنٹ استعمال کی جائے تو ''الیگل فنکشن کال''("Illegal function call") کا ایرر میسج (Error Message) ظاہر ہوتا ہے۔	SCREEN 2
ان تمام موڈز میں بارڈر کلر نہیں دیا جا سکتا۔ گرافکس کی بیک گراؤنڈ، بیک گراؤنڈ کلر سے دی جاتی ہے۔	SCREEN 7- SCREEN 10

نوٹ:۔ اگر مخصوص رینج سے باہر کوئی آرگیومنٹ دیا جائے تو اس کا نتیجہ "Illegal function call" کی صورت میں ظاہر ہوتا ہے۔

مثالیں :

درج ذیل مثالیں COLOR سٹیٹمنٹس اور مختلف سکرین موڈز میں ان کے اثرات کو ظاہر کرتی ہیں:

```
SCREEN 0
COLOR 1, 2, 3  'foreground=1, background=2, border=3
SCREEN 1
COLOR 1, 0     'foreground=1, even palette number
COLOR 2, 1     'foreground=2, odd palette number
SCREEN 7
COLOR 3, 5     'foreground=3, background=5
SCREEN 8
COLOR 6, 7     'foreground=6, background=7
SCREEN 9
COLOR 1, 2     'foreground=1, background=2
```

6.2.3 پیلٹ (PALETTE)

یہ سٹیٹمنٹ پہلے سے موجود دو کلرز کے سیٹ میں سے ایک سیٹ کو منتخب کرنے کے لیے استعمال ہوتی ہے۔ یہ کلر سیٹ LINE, CIRCLE, PSET, DRAW یا دیگر گرافکس یوٹیلیٹیز (Utilities) کے کلر پیرامیٹر میں استعمال ہوتے ہیں۔

Syntax سینٹیکس

PALETTE [attribute, color]

PALETTE سٹیٹمنٹ صرف ایسے سسٹمز میں کام کرتی ہے جن کے ساتھ EGA یعنی Enhanced Graphics Adapter منسلک ہوتا ہے۔ GW-BASIC کا پیلٹ کلرز کے سیٹ پر مشتمل ہوتا ہے جبکہ ہر ایک کلر کا تعین ایک ایٹریبیوٹ سے کیا جاتا ہے۔ ہر ایک ایٹریبیوٹ کے ساتھ ایک اصل ڈسپلے کلر ہوتا ہے (دیکھیں ٹیبل 6.1)۔ اس کلر سے سکرین پر نظر آنے والا اصل کلر متعین ہوتا

ہےاوراس کادارومدارآپ کےسکرین موڈپراورہارڈویئر کی کوالٹی پر ہے۔

پیلٹ کی ویلیو0ہوسکتی ہے یا 1۔اگرپیلٹ 0یا1ہوتوکلرزیہ ہوں گے۔

Palette پیلٹ	Numbers نمبرز	Color رنگ
0	0	Background Color بیک گراؤنڈکلر
0	1	Green سبز
0	2	Red سرخ
0	3	Brown براؤن
1	0	Background Color بیک گراؤنڈکلر
1	1	Cyan سائن
1	2	Magenta میجنٹا
1	3	White سفید

6.3 PSET سٹیٹمنٹ (PSET Statement)

مقصد (Purpose)

گرافکس موڈ کے استعمال کے دوران ایک پوائنٹ کوسکرین پرایک مخصوص جگہ پر دکھانے کے لیے۔

کوآرڈینیٹ ویلیوز (Coordinate Values)

کوآرڈینیٹ کی ویلیوز سکرین کے کناروں سے باہر ہوسکتی ہیں۔تاہم،انٹیجر رینج (32768-تا32767) سے باہر کی ویلیوز کی وجہ سے ''اوورفلو(Overflow)'' کی ایرر کامیسج آتا ہے۔ہائی-ریزولوشن اور میڈیم-ریزولوشن میں(0,0) ہمیشہ اوپر کا بائیں والا کونہ ہوتا ہےاور (0,199) نیچے کا بائیں والا کونہ ہوتا ہے۔اگرکلر کی ویلیو3 سے بڑی ہوتو ''الیگل فنکشن کال'' کی ایرر کامیسج آتا ہے۔

Syntax

سینٹیکس

PSET(x,y)[,color]

PSET (x offset, y offset) موجودہ پوائنٹ کے لحاظ سے ایک پوائنٹ ہے۔مثال کے طور پر:

```
PSET (10,10)
```

مثال1-

```
40 FOR I= 100 TO 0 STEP-1
50 PSET(I,I),0
60 NEXT I
```

6.4 LINE سٹیٹمنٹ (Line Statement)

LINE سٹیٹمنٹ کا کام سکرین پرلائنیں اور باکس بنانا ہے۔ہم کوئی سی بھی دو سٹیٹمنٹس کے درمیان لائن لگانے کے لیے LINE سٹیٹمنٹ استعمال کرسکتے ہیں۔

Syntax سینٹیکس

LINE [(x1,y1)]-(x2,y2) [,[attribute][,B[F]][,style]]

سینٹیکس میں ویلیوز (x1, y1,) اور (x2, y2) بالترتیب لائن کے شروع کے اور آخری نقطہ کے مقامات کی نشاندہی کے لیے استعمال ہوتی ہیں۔ان دونوں پوائنٹس کی پوزیشن کو نفی (minus) یا ڈیش کی علامت سے جدا کیا جاتا ہے۔ایٹریبیوٹ سے ڈسپلے پکسل کے کلر یا گہرائی کا پتہ چلتا ہے۔

بی باکس B(box) پوائنٹس (x1, y1) اور (x2, y2) کے متضاد کناروں پر باکس بنانے کے لیے استعمال ہوتا ہے۔ BF(filled box) سے B کی طرح باکس بنتا ہے اور یہ اندر سے پوائنٹس کی مدد سے فل (Fill) کر دیا جاتا ہے۔سٹائل (Style) ایک 16 بٹ انٹیجر ماسک (Mask) ہے جو سکرین پر پکسلز بنانے کے لیے استعمال ہوتا ہے۔اسے لائن سٹائلنگ کہتے ہیں۔سٹائل 0، 1، 2، 3، 4 اور 5 میں سے کوئی ایک ہو سکتا ہے۔یہ نارمل لائنوں اور باکسز کے لیے استعمال ہوتا ہے لیکن فلڈ (Filled) باکسز میں استعمال نہیں ہوتا۔

LINE کی سادہ ترین شکل درج ذیل ہے۔

LINE -(x2,y2)

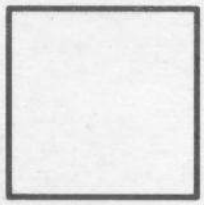
یہ موجودہ پوائنٹ سے (x2,y2) تک ایک لائن بناتی ہے جس میں فورگراؤنڈ کلر استعمال ہوتا ہے۔

مثال 1۔

```
LINE (160, 0)- (160, 199)
```



```
LINE (0,0)-(100,175),,B
```



سکرین کے اوپر کے بائیں کونے میں مربع شکل کا ایک باکس

6.5 سرکل سٹیٹمنٹ (CIRCLE Statement)

CIRCLE سٹیٹمنٹ کا کام گرافکس موڈ میں سکرین پر دائرہ یا بیضوی شکل بنانا ہے۔

Syntax سینٹیکس

CIRCLE(x, y), radius[,[color][,[start],[end][,aspect]]]

(x,y) سے مراد ایکس ایکسز (x-axis) اور وائے ایکسز (y-axis) ہیں جو کہ دائرے یا بیضوی شکل کے مرکزہ کے کوآرڈینیٹس ہیں۔

ریڈیئس radius سے مراد دائرے یا بیضوی شکل کا رداس (radius) ہے۔x اور y مقداریں ایکسپریشنز کی شکل میں ہوسکتی ہیں۔کلر (color) سے مراد دائرے یا بیضوی شکل کا کلر ہے۔اس کی قیمت کا دارومدار موجودہ سکرین موڈ پر ہے۔ہائی ریزولوشن موڈ میں 0 سیاہ رنگ کو اور 1 سفید رنگ کو ظاہر کرتے ہیں۔آغاز (start) اور اختتام (end) کے زاویہ کے پیرامیٹرز 2π – اور 2π کے درمیان دائروی مقداریں ہیں جو بتاتی ہیں کہ بیضوی شکل کی ڈرائنگ کہاں سے شروع ہوگی اور کہاں پر ختم ہوگی۔

aspect کا پیرامیٹر x-axis اور y-axis میں نسبت (x:y) ظاہر کرتا ہے۔موجودہ نسبت کا دارومدار سکرین موڈ پر ہے۔

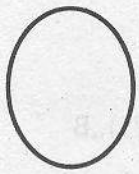
مثال 1۔

```
10 SCREEN1: CIRCLE(100,100), 50
```

بیضوی شکل ایک ریاضیاتی اصطلاح ہے جو کہ بیضہ یعنی انڈے جیسی شکل کے لیے استعمال ہوتی ہے۔ CIRCLE سٹیٹمنٹ میں ایک پیرامیٹر کے اضافے سے بیضوی شکل بنائی جاسکتی ہے۔

مثال 2۔

```
10    REM  draw  an ellipses
20    CLS
30    SCREEN 2
40    CIRCLE  (40,80),30,1,,,1
50    END
```



اس صورت میں بیضوی شکل کی اونچائی اس کی چوڑائی سے زیادہ ہوگی۔

مثال 3۔

```
10    REM  draw  an ellipse
20    CLS
30    SCREEN 2
40    CIRCLE  (40,80),30,1,,,2
50    END
```

مندرجہ بالا پروگرام سے بننے والی بیضوی شکل کی چوڑائی اوپر والی شکل سے قدرے کم ہوگی۔

6.6 DRAW سٹیٹمنٹ (DRAW Statement)

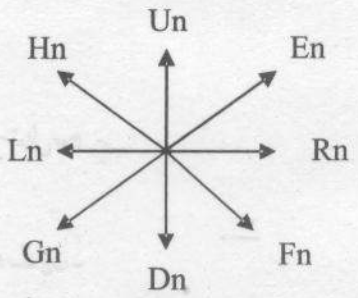
DRAW سٹیٹمنٹ سکرین پر لائنیں اور شکلیں بنانے کے لیے استعمال ہوتی ہے۔ یہ صرف گرافکس موڈ میں کام کرتی ہے۔ DRAW سٹیٹمنٹ دائروی شکل کے علاوہ کوئی اور شکل بنانے کے لیے استعمال ہوتی ہے۔ یہ شکل سٹیٹمنٹ کے ساتھ دیے گئے سٹرنگ کے مطابق بنتی ہے۔

Syntax سینٹیکس

DRAW string

سٹرنگ ایک حرف کی کمانڈ پر مشتمل ہوتا ہے جس کے بعد ایک پری فکس (Prefix) آتا ہے جو کہ لائن کے سائز، سمت وغیرہ کو کنٹرول کرتا ہے اور کوٹیشن مارکس میں دیا گیا ہوتا ہے۔ حرکت دینے والی درج ذیل کمانڈز میں سے ہر ایک موجودہ گرافکس پوزیشن سے حرکت دیتی ہے۔ حرکت دینے والی کمانڈز سکیل فیکٹر n* کے مطابق حرکت دیتی ہیں، جبکہ n کی ڈیفالٹ ویلیو 1 ہے؛ اس لیے، اگر n نہ دیا جائے تو وہ ایک پوائنٹ آگے حرکت دیتی ہیں اور سکیل فیکٹر کی موجودہ ویلیو استعمال کرتی ہیں۔

کمانڈ	Moves	حرکات
Un	up	اوپر
Dn	down	نیچے
Ln	left	بائیں
Rn	right	دائیں
En	diagonally up and right	ترچھے رُخ اوپر اور دائیں
Fn	diagonally down and right	ترچھے رُخ نیچے اور دائیں
Gn	diagonally down and left	ترچھے رُخ نیچے اور بائیں
Hn	diagonally up and left	ترچھے رُخ اوپر اور بائیں

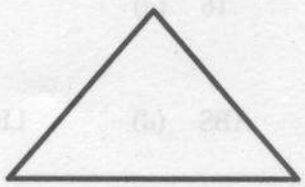


مندرجہ ذیل پری فکس کمانڈز اوپر والی حرکت کرنے والی کمانڈز پر فوقیت رکھ سکتی ہیں۔

B موو، لیکن کوئی پوائنٹ نہ بنے۔

N موو، لیکن جب کام ختم ہو جائے تو اور یجنل پوزیشن پر واپس آ جائیں۔

```
10  REM PROGRAM TO DRAW A TRIANGLE
20  SCREEN 2
30  PSET(250,50)
40  DRAW "G50 R100 H50"
```



```
50    REM PROGRAM TO DRAW A SQUARE
60    SCREEN 2
70    PSET(250,50)
80    DRAW "R50 D50 L50 U50"
```

مشق

1- خالی جگہیں پر کریں۔

(i) SCREEN 2 ---------- ریزولوشن گرافک موڈ ہے۔

(ii) ٹیکسٹ موڈ میں سکرین پر کریکٹرز ---------- کالموں اور ---------- قطاروں میں ظاہر کیے جاتے ہیں۔

(iii) میڈیم۔ ریزولوشن موڈ میں سکرین کو ---------- پکسل کے میٹرکس میں تقسیم کیا جاتا ہے۔

(iv) ---------- سٹیٹمنٹ میں ایک سٹرنگ ڈبل کریکٹر کمانڈز پر مشتمل ہوتا ہے جس کے بعد سائز اور سمت کنٹرول کرنے والا پری فکس آتا ہے۔

(v) ---------- سٹیٹمنٹ لائنیں اور دیگر اشکال بنانے کے لیے استعمال ہوتی ہے۔

(vi) موڈ 3 میں پیلٹ 0 سے تعلق رکھنے والا کلر ---------- ہے۔

(vii) ہائی۔ ریزولوشن موڈ میں ---------- قطاروں اور ---------- کالموں کا میٹرکس استعمال ہوتا ہے۔

(viii) ایک پکچر عمدہ ڈاٹس سے بنی ہوتی ہے جو کہ ---------- کہلاتے ہیں۔

(ix) پیلٹس کے نمبر ---------- اور ---------- ہوتے ہیں جبکہ ہر ایک کے چار کلرز ہوتے ہیں۔

(x) کلر سٹیٹمنٹ ---------- ریزولوشن گرافک موڈ میں کام نہیں کرے گی۔

2- درست جواب کا انتخاب کریں۔

(i) ان میں سے کون سی درست گرافک موڈ کمانڈ نہیں ہے؟

LINE (a) PSET (b) COLOR (c) (d) ان میں سے کوئی نہیں

(ii) بیسک میں موجود بیک گراؤنڈ کلرز کی تعداد ہے:

4 (a) 8 (b) 12 (c) 16 (d)

(iii) کون سا فنکشن بیسک میں آؤٹ پٹ دینے کے طریقے کو تبدیل کرتا ہے؟

SCREEN (a) PRINT (b) LEFT$ (c) ABS (d)

(iv) کلر مونیٹرز پر SCREEN موڈز کی اقسام ہیں:

2 (a) 3 (b) 4 (c) 5 (d)

(v) عام طور پر کمپیوٹر مونیٹر کی سکرین ہوتی ہے:

(a) گرافک موڈ میں (b) ٹیکسٹ موڈ میں

(c) (a) اور (b) (d) ان میں سے کوئی نہیں

(vi) درج ذیل سٹیٹمنٹ سے لائن کھینچی جاسکتی ہے:

(a) LINE (b) DRAW (c) a اور b دونوں (d) ان میں سے کوئی نہیں

(vii) DRAW سٹیٹمنٹ میں شروع کا حرف "B" استعمال ہوتا ہے:

(a) بلاک بنانے کے لیے (b) بلیک مارک کے ساتھ حرکت کرنے کے لیے

(c) (a) اور (b) (d) بلیو مارک کے ساتھ حرکت کرنے کے لیے

(viii) سرخ، سبز اور براؤن رنگوں کا سیٹ دیا گیا ہے:

(a) پیلٹ 0 میں (b) پیلٹ 1 میں (c) (a) اور (b) (d) ان میں سے کوئی نہیں

(ix) CIRCLE سٹیٹمنٹ کی مدد سے ہم مزید کھینچ سکتے ہیں:

(a) لائن (b) باکس (c) بیضوی شکل (d) PSET

(x) میڈیم۔ ریزولوشن سکرین موڈ میں فور گراؤنڈ کے کلرز ہیں:

(a) 1 (b) 2 (c) 4 (d) 16

3- غلط اور صحیح بیانات کی نشاندہی کریں۔

(i) کسی ایک پکسل کو دیکھنا اور اس کی پیائش کرنا مشکل ہے۔

(ii) COLOR سٹیٹمنٹ کی فارمیٹ کا دارومدار ٹیکسٹ موڈ پر ہے۔

(iii) ٹیکسٹ موڈ سے ہم اچھی تصویر نہیں بنا سکتے۔

(iv) PSET سٹیٹمنٹ کو آرڈی نیٹس x،y بیان کرنے کے لیے استعمال ہوتی ہے۔

(v) DRAW نچلے درجے کے منطقی گرافک پروگرام کے طور پر استعمال ہوتی ہے۔

(vi) LINE سٹیٹمنٹ کے ساتھ کو آرڈی نیٹس کے دو جوڑے استعمال ہوتے ہیں۔

(vii) LINE سٹیٹمنٹ سکرین پر ترچھی لائن بنا سکتی ہے۔

(viii) SCREEN فنکشن تین میں سے کسی ایک موڈ پر سیٹ کیا جاسکتا ہے۔

(ix) ٹیکسٹ موڈ میں کریکٹرز سکرین پر 40 کالموں اور 25 قطاروں میں ظاہر کیے جاتے ہیں۔

(x) CIRCLE سٹیٹمنٹ بیضوی شکل بنانے کے لیے بھی استعمال ہوسکتی ہے۔

4- گرافک کی تعریف بیان کریں۔ گرافک موڈز کے نام بتائیں۔

5- ٹیکسٹ موڈ، میڈیم۔ ریزولوشن موڈ اور ہائی۔ ریزولوشن موڈ کے کو آرڈی نیٹس کیا ہیں؟

-6 SCREEN سٹیٹمنٹ کی تعریف بیان کریں۔

-7 CIRCLE سٹیٹمنٹ کا سینٹیکس تحریر کریں۔ وضاحت کے لیے مثال بھی تحریر کریں۔

-8 LINE اور DRAW سٹیٹمنٹس کا موازنہ کریں اور ان میں فرق بیان کریں۔

-9 درج ذیل سٹیٹمنٹس میں اگر کوئی ایررز ہیں تو نشاندہی کریں۔

a).
```
LINE (140,100)-(300-100),2,BF,4
```

b).
```
10  SCREEN 2
20  COLOR 1, 2
30  DRAW "U10 R10 D10 L10"
```

c).
```
10  SCREEN 1
20  A=20
30  DRAW "U=A R=A, D=A L=A"
```

-10 درج ذیل کا آؤٹ پٹ کیا ہوگا؟

a).
```
10  SCREEN 2
20  PSET(250, 50)
30  DRAW "G50 R100 H50"
40  END
```

b).
```
10  SCREEN 2
20  FOR I = 30 TO 180
30  CIRCLE(1,100), 50
40  NEXT I
```

-11 ستارہ (star) بنانے کے لیے پروگرام تحریر کریں۔

-12 COLOR سٹیٹمنٹ کی تعریف بیان کریں۔ کلر سٹیٹمنٹ سے کتنی کلر بیک گراؤنڈز میسر ہوتی ہیں؟

-13 پیلٹ (Palette) کی تعریف بیان کریں۔

-14 مختصر طور پر پکسل (Pixel) پر بحث کریں۔

-15 مختلف رداس کے پانچ ہم مرکز دائرے بنانے کے لیے پروگرام تحریر کریں۔

-16 DRAW سٹیٹمنٹ استعمال کرتے ہوئے متوازی الاضلاع بنانے کے لیے پروگرام تحریر کریں۔

جوابات

-1

(i)	ہائی	(ii)	40, 25	(iii)	300×200	(iv)	DRAW	(v)	DRAW
(vi)	براؤن	(vii)	640, 200	(viii)	پکسلز	(ix)	0, 1	(x)	ہائی

-2

(i)	c	(ii)	d	(iii)	a	(iv)	d	(v)	b
(vi)	c	(vii)	c	(viii)	a	(ix)	c	(x)	a

-3

(i)	صحیح	(ii)	غلط	(iii)	صحیح	(iv)	صحیح	(v)	غلط
(vi)	صحیح	(vii)	صحیح	(viii)	غلط	(ix)	صحیح	(x)	صحیح

باب 7

مائیکروسوفٹ ورڈ

(MICROSOFT WORD)

7.1 ورڈ پروسیسنگ سے تعارف (Introduction to Word Processing)

شروع میں ماضی کے لوگ پین یا قلم اور کاغذ کی مدد سے ڈاکیومنٹ تحریر کیا کرتے تھے۔ پھر اس کے بعد انہوں نے ٹائپ رائٹر استعمال کرنا شروع کر دیا لیکن لوگوں کو پھر بھی بہت سے مسائل درپیش تھے جیسا کہ ڈاکیومنٹ مٹانا یا اس کی درست کرنا، وغیرہ۔ کمپیوٹر سوفٹ ویئر نے یہ مسائل حل کر دیے۔ کئی قسم کے سوفٹ ویئر ہیں جو کہ خاص کام سرنجام دیتے ہیں۔ مثال کے طور پر ورڈ پروسیسر ایک ایپلیکیشن سوفٹ ویئر (Application software) ہے جو کہ ڈاکیومنٹ تحریر کرنے کے لیے استعمال ہوتا ہے۔ مائیکروسوفٹ ورڈ ایک طاقتور ورڈ پروسیسنگ پروگرام ہے۔

مائیکروسوفٹ ورڈ (MS Word)، ڈاکیومنٹس بنانے کے لیے ایک بے حد ضروری ٹول ہے۔ اسے استعمال کرنے کی آسانی نے اسے مارکیٹ میں موجود سب سے زیادہ استعمال ہونے والی ورڈ پروسیسنگ ایپلیکیشن بنا دیا ہے۔ آپ اسے پروفیشنل لنکنگ (Professional linking) ڈاکیومنٹس تحریر کرنے کے لیے استعمال کر سکتے ہیں۔ اگر آپ دوست کو ایک سادہ سا خط تحریر کرنا چاہتے ہیں یا آپ تفصیلی، کئی صفحات پر مبنی رپورٹ لکھنا چاہتے ہیں جس میں گرافکس اور عددی ڈیٹا پر مشتمل ٹیبلز ہوں تو آپ یہ کام تیزی سے اور آسانی سے مائیکروسوفٹ ورڈ میں کر سکتے ہیں۔ یہ ایک ہی پروگرام میں ٹیکسٹ، سپریڈ شیٹس (Spread sheets) اور گرافکس کو اکٹھا کرنے کی سہولت دیتا ہے۔ آپ ورڈ کی مدد سے اپنے ویب پیجز (Web pages) بھی بنا سکتے ہیں۔ ورڈ کے کئی قسم کے مینیوز، ٹول بارز (Toolbars) اور بٹنز (Buttons) کی وجہ سے اسے سیکھنا اور استعمال کرنا بہت آسان ہو گیا ہے۔ ایم ایس ورڈ کی خصوصیات درج ذیل ہیں:

- تیزی سے اور آسانی سے ڈاکیومنٹس تخلیق کرنا۔
- لیٹرز (Letters)، جرنلز (Journals)، فیکس (Fax) پیغامات، کیلنڈرز اور دیگر بہت سی اقسام کے ڈاکیومنٹس بنانے کے لیے بلٹ-اِن ٹیمپلیٹس (Templetes) مہیا کرنا۔
- بلٹ-اِن سپیل چیکر (Spell checker) اور گرامر مہیا کرنا۔
- صرف ماؤس کلک (Click) سے آٹو کوریکشن (Auto correction) کی سہولت مہیا کرنا۔
- آٹو ٹیکسٹ اینٹری کی سہولت دینا تاکہ شارٹ-کی کی مدد سے طویل ٹیکسٹ یا پیراگراف انسرٹ (Insert) کیا جا سکے۔
- کاغذ پر ایک ڈاکیومنٹ کی کئی نقول فراہم کرنا۔
- موجودہ ڈاکیومنٹ کھولنے کی سہولت مہیا کرنا۔
- آئندہ استعمال کے لیے ڈاکیومنٹ کو محفوظ کرنا۔

- آسانی سے ایررز درست کرنے اور فارمیٹ کرنے کی سہولت مہیا کرنا۔
- ہم نے جو کچھ ٹائپ کیا ہے یا جو کام کیا ہے اسے اَن ڈُو (Undo) یا ری ڈُو (Redo) کرنے کی سہولت مہیا کرنا۔
- تیزی سے ٹیکسٹ فائنڈ (Find) کرنے یا تبدیل کرنے کی سہولت مہیا کرنا۔
- ٹیبلز، پکچرز بنانے اور فارمولے استعمال کرنے کی سہولت مہیا کرنا۔
- تمام بہترین خصوصیات میں سے ایک خصوصیت ڈاکیومنٹ کو HTML میں محفوظ کرنا ہے۔

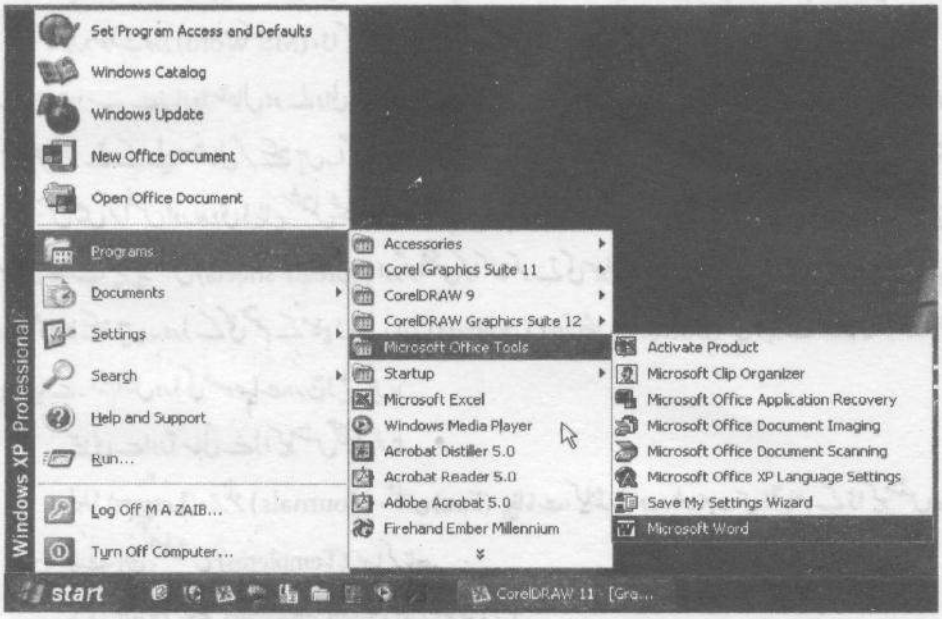
نوٹ: ہوسکتا ہے کہ ان میں سے بعض خصوصیات ایم ایس ورڈ کے پرانے ورژنز میں میسر نہ ہوں۔

7.2 مائیکروسوفٹ ورڈ لوڈ کرنا (Loading Microsoft Word)

مائیکروسوفٹ ورڈ لوڈ (Load) کرنے سے مراد کمپیوٹر پر ورڈ پروگرام چلانا ہے۔ اس کی کمانڈ کی ترتیب یہ ہے:

"Start" >> "Programs" >> "Microsoft Office" >> "Microsoft Office Word"

اگر ڈیسک ٹاپ (Desktop) پر مائیکروسوفٹ ورڈ کا (مربع شکل کا آئیکن (Icon) جس کے درمیان میں W لکھا ہو) موجود ہو تو آپ اسے ڈبل (Double) کلک کرکے بھی پروگرام کھول سکتے ہیں۔

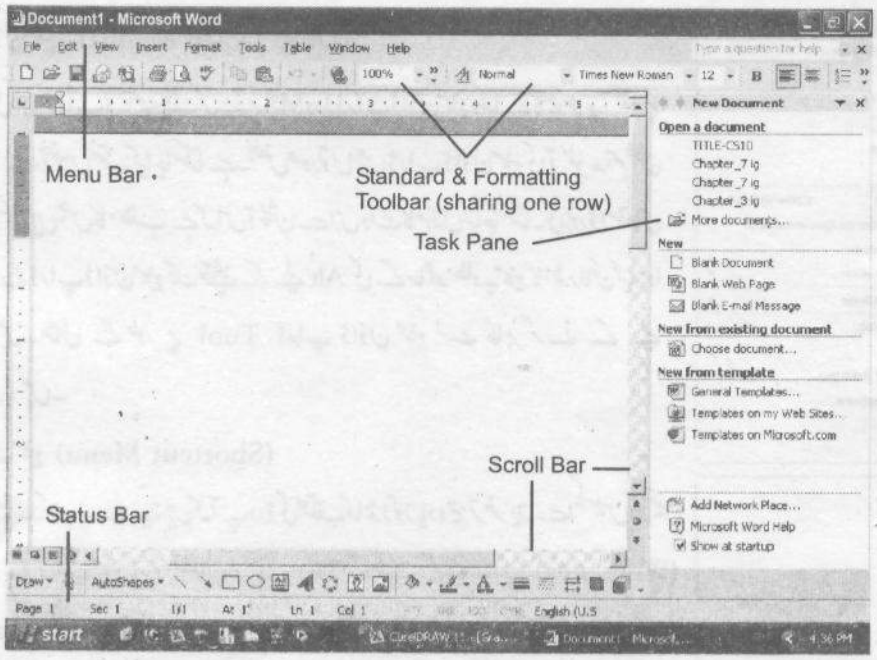


MS پروگرام لوڈ کرنے کے بعد اسے ڈاکیومنٹ بنانے، ایڈٹ (Edit) کرنے اور محفوظ کرنے کے لیے استعمال کیا جاسکتا ہے۔

سکرین لے آؤٹ (Screen Layout)

ورڈ ڈاکیومنٹ کے لیے ایپلیکیشن ونڈو جگہ مُہیا کرتی ہے۔ ایپلیکیشن ونڈو کے مختلف حصے مندرجہ ذیل ہیں:

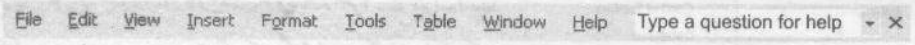
- ٹائٹل بار
- مینیو بار
- ٹُول بار
- ڈاکیومنٹ ونڈو
- سٹیٹس بار
- ویو بٹن



ٹائٹل بار (Title Bar)

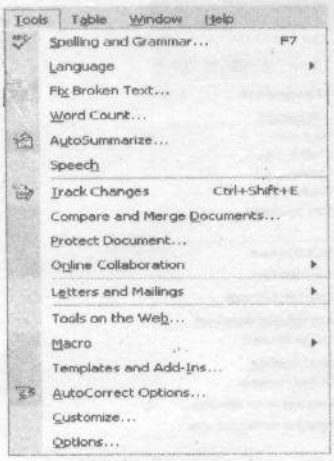
ہم ٹائٹل بار سے شروع کرتے ہیں جو کہ سکرین کے سب سے اوپر والے حصہ میں ہوتا ہے۔ ٹائٹل بار میں، مائیکروسوفٹ ورڈ اُس ڈاکیومنٹ کا نام دکھاتا ہے جس پر آپ اِس وقت کام کر رہے ہوتے ہیں۔ آپ اپنی سکرین کے اوپر والے حصہ میں "Microsoft Word Document1" یا اِسی طرح کا کوئی نام دیکھتے ہیں۔

مینیو بار اور ڈراپ ڈاؤن مینیوز (Menu Bar and Drop Down Menus)



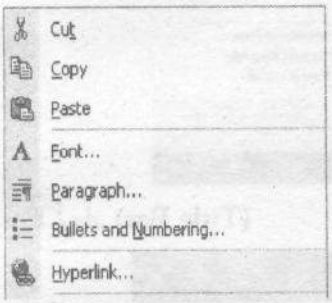
عام طور پر مینیو بار، ٹائٹل بار کے بالکل نیچے پایا جاتا ہے۔ مینیو بار، مینیو دکھاتا ہے۔ مینیو بار File کے لفظ سے شروع ہوتا ہے اور Edit، View، Insert، Format، Tools، Table، Window اور Help تک جاتا ہے۔ آپ سوفٹ ویئر کو ہدایات دینے کے لیے مینیوز استعمال کرتے ہیں۔ اپنا ماؤس ایک مینیو آپشن کی طرف کریں اور بائیں بٹن کو کلک کریں تا کہ ایک ڈراپ۔ ڈاؤن مینیو کھل سکے۔ اب آپ کی بورڈ کی دائیں اور بائیں ایروکیز (Arrow keys) کو استعمال کر سکتے ہیں تا کہ مینیو بار کے مختلف اوپشنز میں دائیں بائیں حرکت کر سکیں۔ آپ اوپر نیچے کی ایروکیز کو استعمال کر سکتے ہیں تا کہ ڈراپ۔ ڈاؤن مینیو میں اوپر نیچے حرکت کر سکیں۔

جب آپ ورڈ میں کام کرنا شروع کریں گے تو آپ اس کے پرانے ورژن کے لحاظ سے اس کے مینیو کے سٹرکچر میں فرق دیکھیں گے۔ ورڈ مینیو میں صرف آپ کی استعمال کی ہوئی کمانڈز ہی نظر آتی ہیں۔ کسی مینیو کے تمام اوپشنز دیکھنے کے لیے آپ کو مینیو کے نیچے کے ڈبل ایروز پر کلک کرنا چاہیے۔ ٹول مینیو کے نیچے کے ایروز کو ڈبل کلک کرنے کے بعد اس کی مکمل شکل یہ ہوگی۔



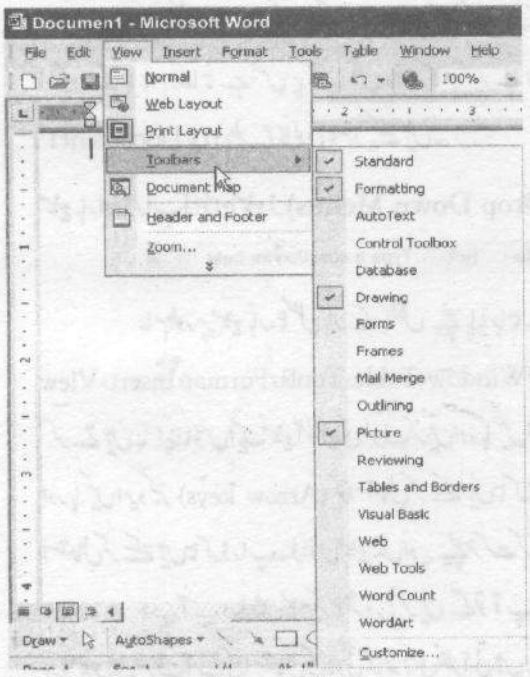
ڈراپ ڈاؤن مینیوز (Drop Down Menus)

مینو بار کی کسی بھی ایک آئیٹم کو کلک کرنے پر MS ورڈ ڈراپ ڈاؤن مینیو دکھائے گا جو مزید آئٹمز پر مشتمل ہوتا ہے، جنہیں منتخب کیا جا سکتا ہے۔ بعض صورتوں میں ڈراپ ڈاؤن مینیو کی آئٹمز مدہم شکل میں ظاہر ہو سکتی ہیں جس کا مطلب ہے کہ اس آپشن سے اس وقت کام نہیں لیا جا سکتا۔ کی بورڈ استعمال کرتے ہوئے ڈراپ ڈاؤن مینیو تک پہنچنے کے لیے Alt کی کے ساتھ مطلوبہ مینیو کا انڈر لائن کیا ہوا حرف دبائیں۔ مثال کے طور پر Tool ڈراپ ڈاؤن مینیو لسٹ ظاہر کرنے کے لیے Alt+T دبائیں۔



شارٹ کٹ مینیو (Shortcut Menu)

یہ فیچرز آپ کو اجازت دیتے ہیں کہ آپ ورڈ کی مختلف کمانڈز کو زیادہ تیز تر طریقہ سے ایکسیس کر سکیں بجائے اس کے کہ آپ مینیو بار کو استعمال کریں۔ ورڈ ڈاکیومنٹ کے ورک ایریا (Work area) میں ماؤس کو رائٹ کلک کرتے ہوئے شارٹ کٹ مینیوز کو دیکھیں۔ اس مینیو کے اوپشنز مختلف ہوں گے اور ان کا انحصار اس بات پر ہوگا کہ آپ نے کس ایلیمنٹ کو رائٹ کلک کیا ہے۔ مثال کے طور پر، مندرجہ ذیل شارٹ کٹ مینیو اُس وقت نظر آتا ہے جب آپ ایک بُلیٹڈ (Bulleted) آئیٹم کو رائٹ کلک کرتے ہیں۔



ٹول بار (Toolbar)

ٹول بارز چھوٹی تصویروں پر مشتمل ہوتے ہیں اور انہیں ٹول آئیکنز کہتے ہیں۔ جب ان کو کلک کیا جاتا ہے تو یہ MS ورڈ میں کام کرنے کا شارٹ کٹ طریقہ مہیا کرتے ہیں۔ MS ورڈ میں 18 ٹول بارز استعمال ہوتے ہیں۔ سکرین پر عام طور پر سٹینڈرڈ اور فارمیٹنگ ٹول بارز نظر آتے ہیں۔ آپ MS ورڈ کو کسٹمائز (Customize) کر کے دیگر ٹول بارز بھی ڈسپلے کر سکتے ہیں۔ زیادہ تیزی اور آسانی سے ایڈیٹنگ اور فارمیٹنگ کے لیے شارٹ کٹ بٹنز پر مشتمل بہت سے ٹول بارز ڈسپلے انگ شارٹ کٹ بٹنز پر مشتمل ہوتے ہیں۔ ٹول بارز منتخب کرنے کے لیے مینیو بار سے View | Toolbars کی کمانڈ منتخب کریں۔ پہلے سے ڈسپلے کیے ہوئے ٹول بارز کے نام کے سامنے چیک مارک (Check mark) (P) لگا ہوتا ہے۔ آپ کسی ٹول بار کے نام پر کلک کر کے اسے مینیو بار کے نیچے دیگر ٹول بارز کے ساتھ ڈسپلے کر سکتے ہیں۔

سٹینڈرڈ ٹول بار (Standard Toolbar)

ورڈ اجازت دیتا ہے کہ تمام ٹول بارز کو اپنی مرضی کے مطابق ڈھالا جائے اس لیے ممکن ہے کہ آپ کو یہاں فہرست میں دیے گئے تمام اوپشنز

نہ مل سکیں۔ بہت سارے ایسے بٹنز ہیں جو کہ فوری طور پر آپ کی ورڈ سکرین پر نظر آسکتے ہیں اور نہیں بھی۔ آپ مندرجہ ذیل گرافکس کو ایک سٹینڈرڈ ٹول بار کے لیے گائیڈ کے طور پر استعمال کر سکتے ہیں۔

(i) **New Blank Documant**: ایک نیا ڈاکیومنٹ شروع کرنے کے لیے، بلینک شیٹ (Blank sheet) کی شکل کے بنے ہوئے نیو بلینک ڈاکیومنٹ کے آئیکن پر کلک کریں۔

(ii) **Open**: اس آئیکن پر کلک کرنے سے کمپیوٹر پر پرانا محفوظ کیا ہوا ڈاکیومنٹ اوپن ہو جاتا ہے۔

(iii) **Save**: سیو آئیکن پر کلک کرنے سے، جس موجودہ ڈاکیومنٹ پر آپ کام کر رہے ہیں، وہ محفوظ ہو جاتا ہے۔ اگر آپ ایک ڈاکیومنٹ پہلی دفعہ سیو کر رہے ہیں تو آپ اس آئیکن پر کلک کر سکتے ہیں۔ تاہم اگر آپ موجودہ ڈاکیومنٹ کو ایک نئی فائل میں سیو کرنا چاہتے ہیں تو آپ کو لازمی طور پر مینیو بار پر جانا ہوگا اور File/Save As کو سلیکٹ کرنا ہوگا اور فائل کو نیا نام دینا پڑے گا۔ جب آپ کسی بھی ڈاکیومنٹ پر کام کر رہے ہوں، تو یقینی طور پر آپ کو اکثر اوقات اسے سیو کرتے رہنا چاہیے تا کہ آپ کا کام ضائع نہ ہو۔

(iv) **Permission**: مائیکروسوفٹ کی نئی ورژن میں IRM (Information Rights Management) کی اجازت ہے جس کی مدد سے آپ اپنے خفیہ ڈاکیومنٹس کو با حفاظت رکھ سکتے ہیں تا کہ اُنھیں نہ تو کاپی کیا جا سکے اور نہ ہی فارورڈ کیا جا سکے۔ مزید معلومات اور اوپشنز کے لیے اسے کلک کریں۔ یہ آپشن ورڈ 2000 میں موجود نہیں ہے۔

(v) **Print**: پرنٹ آئیکن پر کلک کرتے ہوئے آپ خود کار طریقہ سے اُس ڈاکیومنٹ کو پرنٹ کر سکتے ہیں جس پر آپ اس وقت کام کر رہے ہوں۔ اگر پرنٹ اوپشنز کے بارے میں زیادہ جاننا ہو تو مینیو بار میں سے "File">>"Print" سلیکٹ کریں۔

(vi) **Print Preview**: پرنٹ کرنے سے پہلے اپنے ڈاکیومنٹس کو دیکھنے کے لیے کہ وہ کیسے پرنٹ ہوں گے آپ اس آئیکن کو کلک کر سکتے ہیں۔

(vii) **Spelling and Grammar**: اس آئیکن پر کلک کرنے سے آپ کے ڈاکیومنٹ کا ایک Review شروع ہو جائے گا تا کہ اس میں سپیلنگ یا گرائمر کی ایررز تلاش کی جا سکیں جن کو صحیح کیا جانا ہو۔

(viii) **Copy**: موجودہ سلیکشن کو کلپ بورڈ پر کاپی کرنے کے لیے جو کہ ڈاکیومنٹس میں کسی اور جگہ پر پیسٹ کیا جا سکتا ہے یا مکمل طور سے ایک الگ پروگرام/ڈاکیومنٹ میں پیسٹ (Paste) کیا جا سکتا ہے۔ ٹیکسٹ کو کاپی کرنے کے لیے Edit/Copy کی کمانڈ دیں۔ Copy بٹن جو کہ ٹول بار پر ہے کو کلک کریں یا CTRL+C کو پریس کریں تا کہ ٹیکسٹ کو کلپ بورڈ (Clip board) پر کاپی کیا جا سکے۔

(ix) **Paste**: پیسٹ بٹن کو کلک کرنے سے کلپ بورڈ میں کاپی کیا ہوا ٹیکسٹ ڈاکیومنٹ میں انسرٹ ہو جاتا ہے۔ پیسٹ کی مدد سے یا تو آپ کاپی شدہ ٹیکسٹ ڈاکیومنٹ میں انسرٹ کر سکتے ہیں یا سلیکٹ کیے ہوئے ٹیکسٹ کو تبدیل کر سکتے ہیں۔

(x) **Undo Typing**: اَن ڈو ٹائپنگ کا بٹن واپس جاتا ہے اور آخری اضافہ یا تبدیلی کو آپ کے ڈاکیومنٹ سے مٹا دیتا ہے۔

(xi) **Insert Hyperlink**: شاید آپ چاہیں کہ کسی خاص ویب سائیٹ (Web site) یا ویب پیج یا کسی اور قسم کی آن لائن فائل کو اپنے ڈاکیومنٹ میں لنک (Link) کریں۔ انسرٹ ہائپر لنک (Hyperlink) بٹن کو استعمال کر کے آپ سلیکٹڈ (Selected) ٹیکسٹ کو ہائپر لنکس میں تبدیل کر سکتے ہیں۔ جب اس آئیکن کو کلک کیا جائے تو ایک نئی ونڈو ظاہر ہوتی ہے جو آپ کو اس پیج کا URL (ویب ایڈریس) انسرٹ کرنے کی اجازت دیتی ہے جسے آپ لنک کرنا چاہتے ہیں۔

(xii) Insert Table: جب اس آئیکن کو کلک کیا جاتا ہے تو ایک چھوٹی سی ونڈو ظاہر ہوتی ہے جو کہ ایک مربع نما گرڈ کی شکل میں ہوتی ہے۔ اس ونڈو کو گائیڈ کے طور پر استعمال کریں تا کہ آپ بتا سکیں کہ آپ کا ٹیبل کتنی قطاروں اور کتنے کالموں پر مشتمل ہوگا۔ قطاروں اور کالموں کی تعداد منتخب کرنے پر ایک ٹیبل خود بخود ورڈ میں ظاہر ہو جاتا ہے۔ ٹیبلز اور بارڈز (Borders) کے بٹن کی مدد سے آپ ٹیبل کو تبدیل کر سکتے ہیں۔ ایک ٹیبل کے ایک خاص حصہ کو تبدیل کرنے کے لیے سلیکٹ کریں یا کرسر کو اس کے اندر رکھیں اور تبدیلیاں جیسا کہ بارڈز اور کلرز اپلائی (Apply) کریں۔

فارمیٹنگ ٹول بار: (The Formatting Toolbar)

ورڈ اجازت دیتا ہے کہ ٹول بارز کو اپنی مرضی کے مطابق ڈھالا جا سکے، اس لیے ممکن ہے کہ آپ کو فہرست میں دیے گئے تمام اوپشنز نہ مل سکیں۔ بہت سارے ایسے بٹنز ہیں جو آپ کے ورڈ کے ورژن میں مل سکتے ہیں اور ہو سکتا ہے کہ ان میں سے کچھ نہ ملیں۔

(i) Style: ورڈ میں سٹائلز (Styles) ٹیکسٹ کے مختلف حصوں کو جلدی سے فارمیٹ کرنے کے لیے استعمال ہو سکتے ہیں۔ مثال کے طور پر کسی ڈاکیومنٹ میں "Normal" یا "Default Paragraph Font" کو ایک ٹیکسٹ کی باڈی کے لیے استعمال کر سکتے ہیں۔ ہیڈنگز کے لیے تین پہلے سے سیٹ کیے ہوئے سٹائلز موجود ہیں۔

(ii) Font: فونٹ ورڈ ڈاکیومنٹ کا ایک آسان لیکن اہم حصہ ہے۔ فونٹ کو چُننا (ٹیکسٹ کا اپنا سٹائل) اس چیز پر اثر انداز ہو سکتا ہے کہ دوسروں کو ڈاکیومنٹ سکرین پر یا پرنٹ کیا ہوا کیسا نظر آتا ہے۔ مثال کے طور پر Arial فونٹ سکرین پر بہتر نظر آتا ہے جبکہ Times New Roman پرنٹ میں زیادہ صاف ہے۔ ٹیکسٹ کو ایک فونٹ اپلائی کرنے کے لیے کرسر سے من پسند ٹیکسٹ کو سلیکٹ کریں اور ڈراپ ڈاؤن مینیو سے ایک فونٹ منتخب کریں۔ جو فونٹ آپ چاہتے ہیں اُس تک جائیں اور اس کو سلیکٹ کریں۔ ایسا کرنے کے لیے اُس نام کو ایک دفعہ ماؤس سے کلک کریں۔ ایک Serif Font (وہ جس کا ایک "Feet" سرکل کیا گیا ہے) اُن پیراگرافز (Pharagraphs) کے لیے Recommend کیا جاتا ہے جن کو کاغذ پر پرنٹ کرنا ہو تا کہ وہ آسانی سے پڑھے جا سکیں۔ مندرجہ ذیل گرافک serif (بائیں طرف کے Times New Roman) اور sans-serif (دائیں طرف کے سیدھے کناروں والے Arial) میں فرق دکھاتا ہے۔ 

(iii) Font Size: بعض اوقات آپ کو کچھ ٹیکسٹ بڑا یا چھوٹا دکھانے کی ضرورت پیش آ سکتی ہے۔ کرسر سے جو ٹیکسٹ چاہیے، اُس کو سیلیکٹ کریں اور ڈراپ ڈاؤن مینیو سے فونٹ سائز منتخب کر کے آپ ٹیکسٹ سائز تبدیل کر سکتے ہیں۔ ٹیکسٹ پیراگرفز کے لیے 10 یا 12 کا فونٹ سائز اچھا ہوتا ہے۔

(iv) Bold: ٹیکسٹ کو بولڈ (Bold) کرنے کے لیے

(v) Italic: ٹیکسٹ کو ترچھا کرنے کے لیے

(vi) Underline: ٹیکسٹ کے نیچے لائن لگانے کے لیے

(vii) Align Left: سکرین/پیپر کے بائیں طرف سے ٹیکسٹ برابر کرنے کے لیے

(viii) **Justify**: سلیکٹ کیے ہوئے ٹیکسٹ کو دائیں اور بائیں طرف سے Align کرنے کے لیے

(ix) **Align Right**: سلیکٹ کیے ہوئے ٹیکسٹ کو سکرین/پیپر کے دائیں جانب Align کرنے کے لیے

(x) **Center**: سلیکٹ کیے ہوئے ٹیکسٹ کو سکرین/پیپر کے درمیان Align کرنے کے لیے

(xi) **Line Space**: لائن کے درمیان فاصلہ کو ایڈجسٹ کرنے کے لیے (سنگل سپیس، ڈبل سپیس وغیرہ دینے کے لیے)

(xii) **Numbering**: ایک نمبرڈ (Numbered) لسٹ بنانے کے لیے

(xiii) **Bullets**: ایک غیر ترتیب شدہ، بلٹیڈ (Bulleted) لسٹ کے لیے

(xiv) **Decrease Indent**: موجودہ سلیکشن کی انڈینٹیشن (Indentation) کو کم کرنے کے لیے (بائیں طرف)

(xv) **Increase Indent**: موجودہ سلیکشن کی انڈینٹیشن کو بڑھانے کے لیے (دائیں طرف)

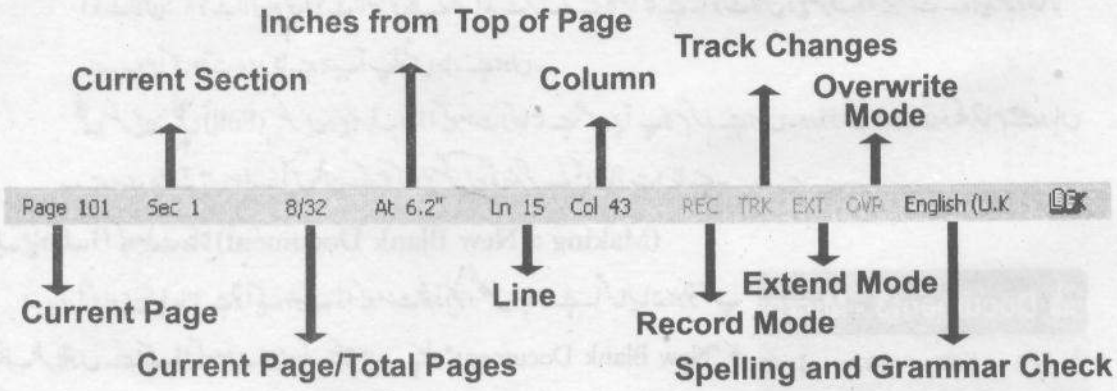
(xvi) **Outside Boder**: موجودہ سلیکشن کے گرد بارڈر بنانے کے لیے۔ ڈراپ ڈاؤن کو کلک کرنے سے بارڈر لگانے کے اوپشنز کی بہت بڑی رینج سامنے آتی ہے۔

(xvii) **Highlight**: موجودہ سلیکشن کو ہائی لائٹ (Highlight) کرنے کے لیے؛ موجودہ کلر پیلا ہے۔

(xviii) **Font Color**: فونٹ کلر کو تبدیل کرنے کے لیے؛ موجودہ/اپنے آپ لگنے والا کلر کالا ہے۔

سٹیٹس بار (Status Bar)

سٹیٹس بار سکرین کے بالکل نیچے آتا ہے اور ایسی انفرمیشن جیسا کہ موجودہ صفحہ، موجودہ سلیکشن، صفحات کی کل تعداد، صفحہ کے اوپر سے فاصلہ، موجودہ لائن نمبر اور موجودہ کالم نمبر مُہیا کرتا ہے۔ سٹیٹس بار ایسے اوپشنز بھی مُہیا کرتا ہے جن کی مدد سے آپ تبدیلیوں کی خبر رکھ سکتے ہیں یا Record Mode، Extension Mode، Overtype Mode اور Spelling اور Grammar Check کو ٹرن آن (Turn on) کر سکتے ہیں۔



سکرول بار (Scroll Bar)

ڈاکیومنٹ ونڈو کے ساتھ سکرول بارز ہوتے ہیں۔ ڈاکیومنٹ ونڈو کی دائیں جانب ایک عمودی سکرول بار ہوتا ہے اور ڈاکیومنٹ ونڈو کے نیچے افقی سکرول بار ہوتا ہے۔ سکرول بارز کی مدد سے ہم ڈاکیومنٹ ونڈو میں اوپر، نیچے، دائیں اور بائیں حرکت کر سکتے ہیں۔

رولر بار (Ruler Bar)

رولر عام طور سے مین ٹول بارز کے نیچے مل جاتا ہے۔ رولر آپ کے ڈاکیومنٹ کا فارمیٹ تیزی سے تبدیل کرنے کے لیے استعمال ہوتا ہے۔ رولر کو ظاہر کرنے کے لیے:

☆ مینیو بار پر View کو کلک کریں۔

☆ رولر کے آپشن کے سامنے ایک چیک مارک ہونا چاہیے۔ اگر اس کے سامنے چیک مارک نہ ہو تو اگلے سٹیپ پر جائیں۔

☆ رولر کو کلک کریں۔ اب رولر ٹول بارز کے نیچے ظاہر ہو جائے گا۔

ویو بٹنز (View Buttons)

ڈاکیومنٹ کے ویو لے آؤٹ کو نارمل (Normal) ویو، ویب لے آؤٹ ویو، پرنٹ لے آؤٹ ویو یا آؤٹ لائن ویو میں تبدیل کرتا ہے۔

ویو کی تبدیلیاں: ویو کی تبدیلیوں سے آپ کو مختلف طریقے مہیا ہوتے ہیں تا کہ آپ اپنے کام کی پروگریس کو دیکھ سکیں اور ٹھیک سے کام کر سکیں۔ ورڈ XP آپ کے ڈاکیومنٹ کو دیکھنے کے پانچ مختلف طریقے مہیا کرتا ہے۔ یہ پانچ ویوز ہیں: نارمل ویو، پرنٹ لے آؤٹ ویو، ویب لے آؤٹ ویو، آؤٹ لائن ویو، اور فُل سکرین ویو۔

نارمل: نارمل ویو ٹائپنگ، ایڈیٹنگ، فارمیٹنگ اور پروف ریڈنگ کے لیے بہت اچھے طریقے سے استعمال کیا جاتا ہے۔ یہ رولرز اور پیج نمبرز کے بغیر آپ کو زیادہ سے زیادہ جگہ مُہیا کرتا ہے۔

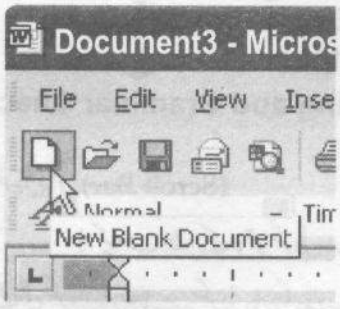
ویب لے آؤٹ: ویب لے آؤٹ ویو آپ کو دکھاتا ہے کہ آپ کا ٹیکسٹ ویب پیج پر کس طرح نظر آئے گا۔

پرنٹ لے آؤٹ: پرنٹ لے آؤٹ ویو آپ کو دکھاتا ہے کہ آپ کا ڈاکیومنٹ پرنٹ ہو کر کیسا نظر آئے گا۔ پرنٹ لے آؤٹ کے اندر آپ پیج کے تمام حصوں کو دیکھ سکتے ہیں۔

آؤٹ لائن: آؤٹ لائن ویو، آؤٹ لائنز کو بنانے اور ایڈٹ کرنے کے کام آتا ہے۔ آؤٹ لائن ویو صرف ڈاکیومنٹ کے ہیڈنگز دکھاتا ہے۔ یہ ویو اُسی وقت مدد دیتا ہے جب آپ نوٹس بنا رہے ہوں۔

فُل سکرین: فُل (Full) سکرین ویو صرف وہ ڈاکیومنٹ دکھاتا ہے جس پر آپ کام کر رہے ہوں۔ ورڈ ڈاکیومنٹ کے دیگر تمام حصے وہاں سے مِٹ جاتے ہیں سوائے اُس بٹن کے جو کہ ویو سکرین کو بند کرنے کی اجازت دیتا ہے۔

ایک نیا بلینک ڈاکیومنٹ بنانا (Making a New Blank Document)



جب ورڈ کو اوپن کیا جاتا ہے تو ایک بلینک ڈاکیومنٹ خود بخود کھل جاتا ہے۔ اگر ایسا نہ ہو تو آپ مختلف طریقوں سے ایک ڈاکیومنٹ اوپن کر سکتے ہیں۔ پہلے "New Blank Document" کا آئیکن ڈھونڈیں، جو کہ ایک خالی شیٹ کی طرح لگتا ہے اور ورڈ کے سٹینڈرڈ مینیو بار کے نیچے ہوتا ہے۔ آئیکن پر کلک کرنے سے ایک نیا خالی ڈاکیومنٹ کھل جاتا ہے۔

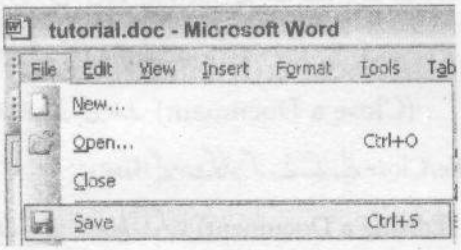
اس کے علاوہ آپ مینیو بار پر جائیں اور File >> New کی کمانڈ سلیکٹ کریں۔

(شارٹ کٹ:N +CTRL)۔ ٹائپنگ شروع کرنے کے لیے صرف کرسر کو بلینک ڈاکیومنٹ کی کسی جگہ پر کلک کریں۔

ایک ڈاکیومنٹ کو اوپن کرنا (Opening a Document)

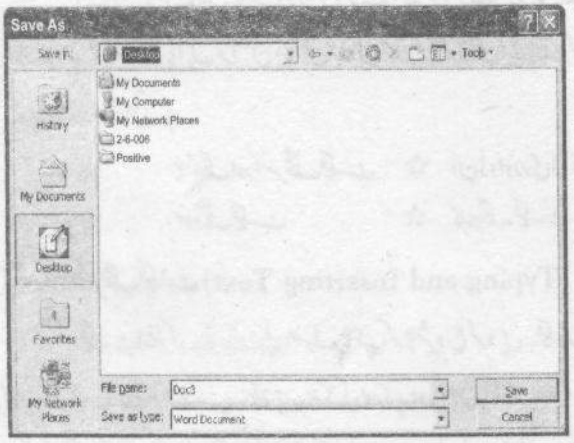
ایک ڈاکیومنٹ کو اوپن کرنے، ویو کرنے، ایڈٹ کرنے یا پرنٹ کرنے کے لیے سب سے پہلے فائل کو ورڈ میں اوپن کرنا پڑتا ہے۔ آپ اوپن فولڈر کے آئیکن کو کلک کر کے ایک فائل کو اوپن کر سکتے ہیں جو کہ سٹینڈرڈ ٹول بار پر پایا جاتا ہے یا آپ مینیو بار استعمال کر سکتے ہیں اور File >> Open (شارٹ کٹ:CTRL+O) کو ڈھونڈ سکتے ہیں۔

ایک ڈاکیومنٹ کو سیو کرنا (Saving a Document)



جب آپ کسی بھی سوفٹ ویئر میں کام کر رہے ہوں تو آپ کو یقین ہونا چاہیے کہ آپ اکثر اوقات اپنے کام کو سیو کرتے ہیں۔ ورڈ میں بے شمار قسم کے اوپشنز ہیں جن کی مدد سے ڈاکیومنٹ کو مختلف قسم کی فائل ٹائپس (Types) میں سیو کیا جا سکتا ہے۔

ایک نئے، غیر محفوظ شدہ (unsaved) ڈاکیومنٹ کو آپ سیو آئیکن کی مدد سے سیو کر سکتے ہیں۔ یہ آئیکن سٹینڈرڈ ٹول بار پر موجود ہوتا ہے۔ اسے کلک کرنے سے ڈاکیومنٹ سیو ہو جاتا ہے۔ اس آئیکن کی شکل فلاپی (Floppy) ڈسک کی طرح ہوتی ہے۔ یا، آپ مینیو بار پر جا کر اور File>>Save کی کمانڈ (جس کا شارٹ کٹ CRL+S ہے) استعمال کر کے ڈاکیومنٹ کو سیو کر سکتے ہیں۔



جب آپ نیا ڈاکیومنٹ بناتے ہیں تو اس کا کوئی کام نہیں ہوتا۔ اگر آپ ڈاکیومنٹ کو بعد میں دیکھنا چاہتے ہیں تو اس کا کوئی نام ہونا چاہیے تا کہ ورڈ اُسے ڈھونڈ سکے۔ جب آپ پہلی مرتبہ ڈاکیومنٹ کو سیو کرتے ہیں تو ورڈ ایک نام پوچھتا ہے اور اس کے بعد جو نام آپ اس کو دیتے ہیں وہ سکرین کے ٹائٹل بار میں ظاہر ہو جاتا ہے۔

ایک ڈائیلاگ باکس (Dialogue box) لازمی طور پر ظاہر ہوتا ہے جو کہ آپ کو بے شمار اوپشنز مہیا کرتا ہے۔ ڈاکیومنٹ کو اپنی پسندیدہ جگہ پر سیو کرنے کے لیے کمپیوٹر پر مطلوبہ فولڈر تلاش کریں اور اِسے سلیکٹ کریں۔ فائل کے نام والے ٹیکسٹ باکس میں اپنے ڈاکیومنٹ کو ایک نام دیں۔ آپ اپنے ڈاکیومنٹ کو لمبے نام بھی دے سکتے ہیں لیکن ایسا نام دیں جو کہ آپ کو یاد رہ سکے۔

پہلے سے محفوظ شدہ کھلے ڈاکیومنٹ کو نئے نام سے محفوظ کرنے کے لیے آپ Save As کا آپشن استعمال کر سکتے ہیں۔ وہ ڈاکیومنٹ اوپن کریں جس کو بالکل نئی فائل میں سیو کرنا چاہتے ہیں۔ مینیو بار پر جائیں، اور File>>Save as کی کمانڈ منتخب کریں۔ فائل نیم (Name) ٹیکسٹ باکس میں اپنے ڈاکیومنٹ کو ایک نیا نام دیں۔ اس اوپشن کو استعمال کرنے سے آپ کو اصل فائل کئی ناموں سے اور مختلف ورژنز میں سیو کرنے کی سہولت مل جاتی ہے۔

نوٹ:۔ سیو اِن (save in) ڈراپ ڈاؤن لسٹ باکس فولڈر اوپشنز کی فہرست دکھاتا ہے جہاں آپ فائل محفوظ کر سکتے ہیں۔ ڈیفالٹ فولڈر جو کہ ظاہر ہوتا ہے اس کا نام "Documents" ہے۔ اگر آپ ڈاکیومنٹ اس فولڈر میں سیو نہیں کرنا چاہتے یا اسے کسی اور ڈسک پر سیو کرنا چاہتے ہیں تو آپ اسے بھی منتخب کر سکتے ہیں۔ براؤز کرنے کے لیے ڈاؤن ایرو پر کلک کریں۔ فائل کے نام میں سپیسز یا سپیشل کریکٹرز استعمال نہ کریں۔ مثال کے طور پر فائل کا لمبا نام اس طرح کا ہو سکتا ہے:

رینیمنگ ڈاکیومنٹس (Renaming Documents)

اس کے ذریعہ پروگرام کو استعمال کرتے ہوئے ایک ورڈ ڈاکیومنٹ کا نام بدلنے کے لیے File/Open کو سلیکٹ کریں اور اُس فائل کو تلاش کریں جس کا نام بدلنا ہو۔ ڈاکیومنٹ کے نام پر رائٹ کلک کریں اور لسٹ میں سے Rename کو سلیکٹ کریں۔ فائل کا نام ٹائپ کرتے ہوئے Enter۔ کی کو پریس کریں۔

کلوزاے ڈاکیومنٹ (Close a Document)

موجودہ ڈاکیومنٹ کو کلوز کرنے کے لیے File/Close کو سلیکٹ کریں یا اگر ٹول بار پر کلوز آئیکن نظر آ رہا ہے تو ماؤس سے اسے کلک کریں۔

ایک ڈاکیومنٹ کو ایڈٹ کرنا (Editing a Document)

ایک ڈاکیومنٹ میں ٹیکسٹ کو داخل کرنے، تبدیل کرنے اور مٹا دینے کے عوامل کو ٹیکسٹ ایڈیٹنگ کہتے ہیں۔ اسی طرح، کونٹینٹس کو داخل کرنے اور تبدیل کرنے یا ایک گرافکس کی شکل تبدیل کرنے کو گرافکس ایڈیٹنگ کرنا کہتے ہیں۔ مندرجہ ذیل لسٹ ورڈ میں عام طور سے استعمال ہونے والے پروسیجرز کو دکھاتی ہے۔

☆ ٹائپنگ اور انسرٹنگ ٹیکسٹ ☆ اَن ڈو اور ری ڈو کمانڈز ☆ سلیکٹنگ ٹیکسٹ ☆ ڈیلیٹنگ ٹیکسٹ
☆ موونگ ٹیکسٹ ☆ کاپینگ ٹیکسٹ ☆ پیسٹنگ ٹیکسٹ

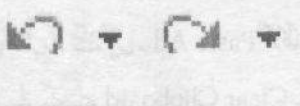
ٹائپنگ اور انسرٹنگ ٹیکسٹ (Typing and Inserting Text)

ٹیکسٹ اینٹر کرنے کے لیے، صرف ٹائپ کرنا شروع کر دیں۔ ٹیکسٹ اُسی جگہ پر ظاہر ہو جائے گا جس جگہ پر کرسر بلنک کر رہا ہے۔ آپ کرسر کو ایرو کیز کی مدد سے حرکت دے سکتے ہیں جو کہ کی بورڈ پر ہوتی ہیں یا ماؤس کو حرکت دے کر اور لیفٹ (Left) بٹن کو کلک کر کے کرسر مطلوبہ جگہ پر لا سکتے ہیں۔ کسی فائل کے ٹیکسٹ میں حرکت کرنے کے لیے مندرجہ ذیل کی بورڈ شارٹ کٹس (Shortcuts) بھی مددگار ثابت ہوتے ہیں۔

موو ایکشن	کی سٹروک
لائن کے شروع میں	HOME
لائن کے آخر پر	END
ڈاکیومنٹ کے شروع میں	CTRL+HOME
ڈاکیومنٹ کے آخر پر	CTRL+END

اَن ڈو اور ری ڈو کمانڈز (The Undo and Redo Commands)

اگر آپ سے اتفاقیہ طور پر ٹیکسٹ ڈیلیٹ ہو جائے یا ٹیکسٹ کو غلط فارمیٹ لگ جائے تو آپ اَن ڈو (Undo) کی کمانڈ استعمال کر کے



Undo Redo

اپنے ڈاکیومنٹ میں کی گئی تبدیلیاں واپس کر سکتے ہیں۔ جب آپ ان ڈو کمانڈ کے بٹن کو کلک کرتے ہیں تو اس سے آپ کا آخری ایکشن واپس ہو جاتا ہے۔ ایک سے زیادہ ایکشنز کو اَن ڈو کرنے کے لیے، اُنھیں اَن ڈو ڈراپ ڈاؤن لسٹ میں سلیکٹ کریں۔ ایک سے زیادہ ایکشنز اُسی ترتیب سے اَن ڈو ہوتے ہیں جس ترتیب سے انھیں کیا گیا ہو۔ زیادہ تر ایکشنز (Actions) واپس ہو سکتے ہیں، لیکن کچھ ایسے بھی ہیں جو کہ واپس نہیں ہو سکتے، جیسا کہ ایک ڈاکیومنٹ کی پرنٹنگ یا سیونگ۔ اگر اَن ڈو کرنے کے بعد آپ اپنا ارادہ بدل دیں تو آپ ری ڈو (Redo) پر کلک کرنے سے اِسے واپس کر سکتے ہیں۔ جب تک کہ آپ مکمل طور سے ٹائپسٹ نہ ہوں، آپ اپنے ڈاکیومنٹ میں کچھ ایررز کر سکتے ہیں یا شاید ڈاکیومنٹ کے کچھ ٹیکسٹ کے بارے میں اپنا مائنڈ تبدیل کر لیں۔ ایک ورڈ پروسیسنگ پروگرام میں یہ تبدیلیاں اور اصلاحات کرنی بہت آسان ہیں۔

ٹیکسٹ کو سلیکٹ کرنا (Selecting Text)

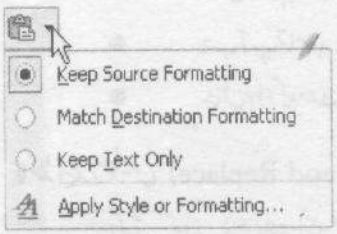
ٹیکسٹ کے کسی ایٹریبیوٹ کو تبدیل کرنے کے لیے سب سے پہلے اس کو ہائی لائٹ کرنا بے حد ضروری ہے۔ ماؤس کو حسب منشا ٹیکسٹ پر رکھتے ہوئے اسے کھینچیں اور اس دوران ماؤس کا بایاں بٹن دبائے رکھیں یا Shift کی کو تھامے ہوئے کی بورڈ پر موجود ایرو کیز کو دبائیں تا کہ ٹیکسٹ ہائی لائٹ ہو جائے۔ مندرجہ ذیل ٹیبل ٹیکسٹ کے ایک پورشن کو سلیکٹ کرنے کے شارٹ کٹ طریقے بتاتا ہے۔

ٹیکنیک	سلیکشن
لفظ پر ڈبل کلک کریں۔	مکمل لفظ
پیراگراف میں تین بار کلک کریں۔	مکمل پیراگراف
الفاظ پر ماؤس ڈریگ (Drag) کریں یا SHIFT کی دبائے رکھیں اور ایرو کیز استعمال کریں۔	کئی الفاظ یا لائنیں
مینو بار سے Edit/Select All کی کمانڈ منتخب کریں یا CTRL+A دبائیں۔	تمام ڈاکیومنٹ

پیج کے کسی بھی حصہ پر، جو کہ سلیکشن کے باہر ہو، پر کلک کرنے سے آپ ٹیکسٹ کو ڈی سلیکٹ کر سکتے ہیں۔ یہی کام آپ کی بورڈ پر ایرو کی کو پریس کر کے کر سکتے ہیں۔

ٹیکسٹ ڈیلیٹنگ (Deleting Text)

Backspace اور Delete کیز، جو کہ کی بورڈ پر واقع ہوتی ہیں، کو استعمال کر کے ٹیکسٹ کو ڈیلیٹ کیا جا سکتا ہے۔ بیک سپیس کرسر کے بائیں جانب کے ٹیکسٹ کو مٹائے گی اور ڈیلیٹ کرسر کے دائیں جانب کے ٹیکسٹ کو مٹائے گی۔ ٹیکسٹ کے ایک بڑے حصہ کو ڈیلیٹ کرنے کے لیے، اسے اوپر بتائے گئے کسی طریقہ کی مدد سے ہائی لائٹ کریں اور Delete کی پریس کریں۔



کلپ بورڈ (The Clipboard)

کوئی بھی ٹیکسٹ جو کہ کٹ یا ڈیلیٹ کیا گیا ہو وہ کلپ بورڈ پر رکھ دیا جاتا ہے۔ آپ کلپ بورڈ کے حصوں کو View/Toolbars/Clipboard جو کہ مینیو بار پر ہے، کی مدد سے دیکھ سکتے ہیں۔

ہر آئیٹم کے کونٹینٹس دیکھنے کے لیے ماؤس کے ایرو کو کلپ بورڈ کے ہر حصہ پر رکھیں۔ کسی حصہ پر کلک کرتے ہوئے اس کی آئیٹمز ڈاکیومنٹ میں شامل کر سکتے ہیں۔ Paste All کو کلک کرنے سے ڈاکیومنٹ میں تمام آئیٹمز کو ایک ساتھ شامل کیا جا سکتا ہے۔ کلپ بورڈ کے حصوں کو صاف کرنے کے لیے Clear Clipboard کے بٹن کو کلک کریں (اس آئیکن پر X کا نشان بنا ہوتا ہے)۔ جب آپ ٹیکسٹ کو کاپی یا پیسٹ کرتے ہیں، تو ایک آئیکن سکرین پر ظاہر ہوتا ہے جو کہ آپ کو آپشن مہیا کرتا ہے کہ ٹیکسٹ کو کس طرح سے پیسٹ کرنا ہے۔

نوٹ:۔ Word XP یوزرز کو Collect اور Paste کی سہولتیں مہیا کرتا ہے جس سے مختلف قسم کی معلومات کو ترتیب دینے کا موقع ملتا ہے۔ مائیکروسوفٹ XP کلپ بورڈ کسی ایک یا زیادہ ڈاکیومنٹس سے، ای میل میسجز سے، ویب پیجز سے، تیار شدہ رپورٹس سے یا دیگر فائلوں سے بارہ عدد ٹیکسٹ کے قطعات یا تصاویر کاپی کرنے کی سہولت مہیا کرتا ہے۔ آپ پھر ان معلومات کو ایک ایک کر کے یا Paste All کی کمانڈ استعمال کرتے ہوئے بیک وقت اپنے ڈاکیومنٹ میں سیو کر سکتے ہیں۔

موونگ (کٹنگ) ٹیکسٹ (Moving (Cutting) Text)

اُس ٹیکسٹ کو ہائی لائٹ کریں جس کو موو کرنا ہے اور مینیو بار میں Edit/Cut کی کمانڈ سلیکٹ کریں یا سٹینڈرڈ ٹول بار پر Cut بٹن کو کلک کریں یا CTRL+X پریس کریں۔ ایسا کرنے سے ٹیکسٹ کلپ بورڈ میں چلا جاتا ہے۔

ٹیکسٹ کے ایک چھوٹے حصہ کو ایک مختصر فاصلہ پر لے جانے کے لیے ڈریگ اینڈ ڈراپ طریقہ زیادہ آسان ہے۔ وہ ٹیکسٹ ہائی لائٹ کریں جسے ایک جگہ سے دوسری جگہ لے جانا ہے، ماؤس سے سلیکشن کو کلک کریں، سلیکشن کو کھینچ کر نئی جگہ پر لے جائیں اور ماؤس بٹن کو چھوڑ دیں۔

ٹیکسٹ کاپینگ (Copying Text)

اُس ٹیکسٹ کو ہائی لائٹ کریں جس کو کاپی کرنا ہے اور مینیو بار میں Edit | Copy کی کمانڈ سلیکٹ کریں یا سٹینڈرڈ ٹول بار پر Copy بٹن کو کلک کریں یا CTRL+C پریس کریں۔ ایسا کرنے سے ٹیکسٹ کلپ بورڈ میں چلا جاتا ہے۔

ٹیکسٹ کو پیسٹ کرنا (Pasting Text)

کٹ کیے ہوئے یا کاپی کیے ہوئے ٹیکسٹ کو پیسٹ کرنے کے لیے کرسر اس جگہ پر لے جائیں جہاں آپ ٹیکسٹ لے جانا چاہتے ہیں اور مینیو بار سے Edit/Paste کی کمانڈ سلیکٹ کریں یا سٹینڈرڈ ٹول بار پر Paste بٹن کو کلک کریں یا CTRL+V پریس کریں۔

- Modify Style ڈائیلاگ باکس سے سٹائل (Style) تبدیل کرنے کے لیے وہی طریقے استعمال کریں جو کہ New Style باکس کے لیے استعمال کیے گئے تھے۔
- سٹائل کا نام بدلنے کے لیے Name کے فیلڈ (Field) میں نیا نام ٹائپ کریں۔
- جب آپ تمام تبدیلیاں کر لیں تو OK پر کلک کریں۔
- سٹائل کو ڈاکیومنٹ میں اپ ڈیٹ کرنے کے لیے Apply پر کلک کریں۔

فائنڈ اینڈ ریپلیس (Find and Replace)

اگر آپ کو ایک خاص لفظ یا کسی ٹیکسٹ کا کوئی حصہ ڈھونڈنا ہو تو آپ Find کی کمانڈ استعمال کر سکتے ہیں۔ اگر آپ پورے ڈاکیومنٹ میں

سے کوئی لفظ ڈھونڈنا چاہتے ہیں تو صرف Find کمانڈ استعمال کریں۔ اگر آپ اپنی (Search) کو ایک مخصوص حصہ تک محدود کرنا چاہتے ہیں تو اُس حصہ کو ہائی لائٹ کریں اور پھر فائنڈ (Find) کمانڈ کو استعمال کریں۔ جب آپ وہ لفظ یا ٹیکسٹ کا کوئی حصہ جس کو کہ آپ ڈھونڈ رہے تھے ڈھونڈ لیں تو آپ اسے نئے ٹیکسٹ کے ساتھ تبدیل کر سکتے ہیں۔ ایسا کرنے کے لیے آپ کو Replace کمانڈ استعمال کرنا پڑے گی۔

فائنڈ - مینیو کے استعمال سے (Find-Using the Menu)

- مندرجہ ذیل کو ٹائپ کریں:

Momi is from Punjab. She lives on the east side of Punjab. Her daughter attends Govt. High School.

- "Momi is from Punjab. She lives on the east side of Pujab. Her daugher attend Govt. High School."

ہائی لائٹ کریں۔

- مینیو میں سے Edit>Find تلاش کریں۔
- Find What کے فیلڈ میں .Govt ٹائپ کریں۔
- Find Next کو کلک کریں۔ نوٹ کریں کہ .Govt ہائی لائٹ کیا ہوا ہے۔
- Find Next کو کلک کریں۔ نوٹ کریں کہ Govt. High School میں .Govt ہائی لائٹ کیا ہوا ہے۔
- Find Next کو کلک کریں۔ مندرجہ ذیل پیغام ظاہر ہوگا:

"Word has finished searching the selection. Do you want to search the remainder of the document?"

- No کو کلک کریں۔
- Cancel کو کلک کریں۔

کیز کے استعمال سے فائنڈ کرنا (Find Using Keys)

- مندرجہ ذیل ٹیکسٹ کو ہائی لائٹ کریں۔

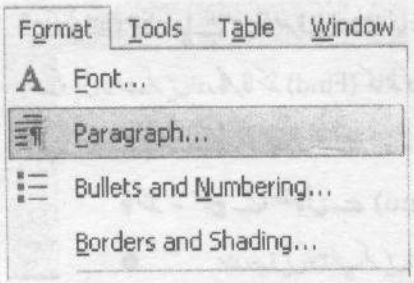
"Momi is from Punjab. She lives on the east side of town. Her daughter attends Govt. High School."

- CTRL+F کو پریس کریں۔
- 5 سے 10 تک سٹپس کو دوھرائیں۔

اسی طرح مینیو اور کیز کے استعمال سے Replace کی کمانڈ پر عمل کیا جا سکتا ہے۔

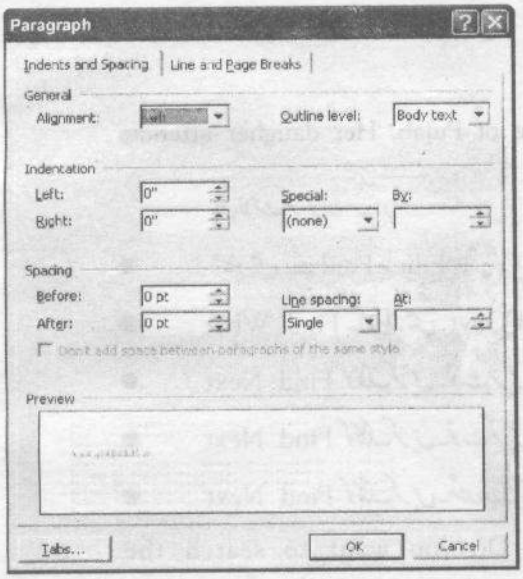
7.3 پیراگراف سپیسنگ (Paragraph Spacing)

پیراگراف فارمیٹنگ کے اوپشنز تک رسائی کے لیے مینیو بار کو نیوی گیٹ کریں اور Format>>Paragraph کو سلیکٹ کریں یا کسی پیراگراف کے اندر رائٹ کلک کریں۔



ایک ونڈو ظاہر ہو جائے گی جس میں سپیس کو تبدیل کرنے اور انڈینٹ کرنے کے تمام اوپشنز ظاہر ہو جائیں گے۔ آپ اس میں سے ڈاکیومنٹ کے ٹیکسٹ کو سنگل یا ڈبل سپیس دے سکتے ہیں اور ڈاکیومنٹ کے مارجنز بھی سیٹ کر سکتے ہیں۔

جب آپ ایک پیراگراف کو فارمیٹ کر رہے ہوں تو آپ کو پورا پیراگراف ہائی لائٹ کرنے کی ضرورت نہیں ہوتی۔ پیراگراف کے اندر کسی بھی جگہ اپنے کرسر کو رکھنے سے آپ اس کو فارمیٹ کر سکتے ہیں۔ ایک پیراگراف کی فارمیٹ سیٹ کر دیں تو باقی آنے والے پیراگرافس بھی اُسی فارمیٹ میں ہوں گے جب تک آپ اُس کا فارمیٹ تبدیل نہ کریں۔



اس سبق کی مشقوں کو حل کرنے کے لیے آپ کو ٹیکسٹ کی ضرورت پڑے گی۔ لہٰذا جو مندرجہ ذیل پیراگراف کو اسی طرح ٹائپ کریں۔ پیراگراف کو اُسی جگہ پر ختم کریں جہاں پر end-of-paragraph کا مارک نظر آ رہا ہے۔ پیراگراف ختم کرنے کے لیے صرف Enter key کو پریس کریں لیکن پیراگراف کے درمیان سپیسز نہ چھوڑیں۔ آپ مشق کے دوران اس سپیس کو سیٹ کریں گے۔ نئی لائن پر جانے کے لیے Enter کو پریس نہ کریں۔ مائیکروسوفٹ ورڈ خود بخود ایک لائن سے دوسری لائن پر چلا جاتا ہے۔

Sample Paragraphs

We will use this paragraph to illustrate several Microsoft Word features. It will be used to illustrate Space Before, Space After, and Line Spacing. Space Before tells Microsoft Word how much space to leave before the paragraph. After tells Microsoft Word how much space to leave after the paragraph. Line spacing sets the space between lines within a paragraph.

We will use this paragraph to illustrate some additional Microsoft Word features. It will be used to illustrate first-line indent. With first-line indent, you can indent the first line of your paragraph. We will also look at indentation. Indentation enables you to indent from the left or right margin of your document.

سپیس بیفور اینڈ سپیس آفٹر (Space Before and Space After)

Space Before پروگرام سے پہلے آنے والے سپیس کی مقدار کو سیٹ کرتا ہے۔ Space After پروگرام سے بعد میں آنے والی جگہ کی مقدار کو سیٹ کرتا ہے۔ مندرجہ ذیل میں Sample Paragraph ہیں جن میں Space After بارہ پوائنٹ (12 pt) سیٹ کیا گیا ہے۔ جو مشقیں آ رہی ہیں اُن میں آپ کو یہ دیکھنے کا چانس (Chance) ملتا ہے کہ Space Before اور Space After کیسے کام کرتی ہیں۔

Space After:

Sample Paragraphs

We will use this paragraph to illustrate several Microsoft Word features. it will be used to illustrate Space Before, Space After, and line spacing. Space Before tells Microsoft Word how much space to leave before the paragraph. Space After tells Microsoft Word how much space to leave after the paragraph. Line spacing sets the space between lines within a paragraph.

We will use this paragraph to illustrate some additional Word features. It will be used to illustrate first-line indent. With first-line indent you can indent the first line of your paragraph. We will also look at Indentation. Indentation enables you to indent from the left and/or right margins of your document.

لائن سپیسنگ (Line Spacing)

لائن سپیسنگ پیراگراف کی لائنز کے درمیان کے سپیس کو سیٹ کرتی ہے۔ پہلے سے موجود لائنز کے درمیان سنگل سپیس ہوتا ہے۔ ہر لائن کے درمیان اُتنی جگہ سیٹ ہوتی ہے جتنا کہ اُسی لائن میں بڑے سے بڑا Font آجائے۔ اگر لائن میں چھوٹے فونٹس ہوں تو وہاں پر لائن کے درمیان اضافی جگہ نظر آتی ہے۔ 1.5 لائنز پر، Line Spacing سنگل لائن کے مقابلہ میں ڈیڑھ گنا زیادہ ہوتی ہے۔ ڈبل سپیڈ لائنز پر Line Spacing سنگل لائن کے مقابلہ میں دوگنا زیادہ ہوتی ہے۔

انڈینٹیشن (Indentation)

انڈینٹیشن کا پیراگراف آپ کو اجازت دیتا ہے کہ آپ اپنے پیراگراف کو بائیں جانب یا دائیں جانب سے انڈینٹ کر سکیں۔ مندرجہ ذیل مثالیں مختلف قسم کی انڈینٹیشن کو دکھاتی ہیں۔

ہم یہ پیراگراف استعمال کر کے ورڈ کے مختلف فیچرز دکھائیں گے۔ ہم Spce Before، Space After اور Line Spacing کی وضاحت کریں گے۔ Space Before ورڈ کو بتاتا ہے کہ پیراگراف سے پہلے کتنی جگہ چھوڑنی ہے۔ Space After ورڈ کو بتاتا ہے کہ پیراگراف کے بعد کتنی جگہ چھوڑنی ہے Line Spacing پیراگراف کی لائنز کے درمیان سپیس کو سیٹ کرتی ہے۔

ہم یہ پیراگراف کچھ اضافی ورڈ فیچرز کی وضاحت کے لیے استعمال کریں گے۔ ہم فرسٹ لائن انڈینٹ کی وضاحت کریں گے۔ Fist Line کا فیچر پیراگراف کی پہلی لائن کو انڈینٹ کرتا ہے۔ ہم Indentation کا بھی جائزہ لیں گے۔ انڈینٹیشن کا آپشن ڈاکیومنٹ کے بائیں یا دائیں مارجنز کو انڈینٹ کرنے کی سہولت دیتا ہے۔

الائنمنٹ (Alignment)

مائیکروسوفٹ ورڈ آپ کو بے شمار قسم کی الائنمنٹس کی چوائس دیتا ہے۔ Left-Justified ٹیکسٹ بائیں جانب الائن ہوتا ہے۔ یہ پہلے سے موجودہ سیٹنگ ہے۔

Sample Paragraph

This is a sample paragraph. It is used to illustrate alignment. Letjustified text is aligned on the left. Right-justified text is aligned on the right. Centered text is centered between the left and right margins. You can use Center to center your titles. Justified text is flush on both sides.

دائیں طرف سے جسٹیفائی کیا ہوا (Right-Justified) ٹیکسٹ دائیں طرف سے برابر ہوتا ہے۔

Sample Paragraph

This is a sample paragraph. It is used to illustrate alignment. Letjustified text is aligned on the left. Right-justified text is aligned on the right. Centered text is centered between the left and right margins. You can use Center to center your titles. Justified text is flush on both sides.

سینٹر رڈ ٹیکسٹ بائیں اور دائیں مارجنز کے درمیان سینٹر ہو جاتا ہے۔

Sample Paragraph

This is a sample paragraph. It is used to illustrate alignment. Letjustified text is aligned on the left. Right-justified text is aligned on the right. Centered text is centered between the left and right margins. You can use Center to center your titles. Justified text is flush on both sides.

جسٹیفائیڈ ٹیکسٹ دونوں طرف سے برابر ہوتا ہے۔

Sample Paragraph

This is a sample paragraph. It is used to illustrate alignment. Letjustified text is aligned on the left. Right-justified text is aligned on the right. Centered text is centered between the left and right margins. You can use Center to center your titles. Justified text is flush on both sides.

دائیں، بائیں، سینٹر سے اور دونوں طرف سے جسٹیفائی کرنے کے کام کی آئیکن اور مینیو کی مدد سے سرانجام دیے جا سکتے ہیں۔ مثال کے طور پر اگر آپ اپنے ورڈ ڈاکیومنٹ کو سینٹر الائن کرنا چاہتے ہیں تو یہ کام پیراگراف الائنمنٹ کے تین مختلف طریقوں سے کیا جا سکتا ہے۔

- کیز کے استعمال سے پیراگراف الائنمنٹ
- آئیکن کے استعمال سے پیراگراف الائنمنٹ
- مینیو کے استعمال سے پیراگراف الائنمنٹ

مینیو کو استعمال کرتے ہوئے سینٹر کرنا۔

- پہلا پیراگراف جو آپ نے ٹائپ کیا تھا اُس کو ہائی لائیٹ کریں جو کہ "We will use" سے شروع ہو کر "within a paragraph." پر ختم ہوتا ہے۔
- مینیو سے Format > Paragraph کی کمانڈ منتخب کریں۔
- انڈینٹس اور سپیسنگ (Indents and Spacing) منتخب کریں۔

- الائنمنٹ کے پل ڈاؤن (Pulldown) مینیو کو کلک کر کے اوپن کریں۔
- سینٹرڈ (Centred) کو کلک کریں۔
- OK کو کلک کریں۔ پیراگراف اب سینٹرڈ ہو چکا ہے۔

کیز کو استعمال کرتے ہوئے جسٹیفائی اور سینٹر کرنا۔

- ٹیکسٹ کو ہائی لائٹ کریں۔
- CTRL+E کو پریس کریں۔ ٹیکسٹ اب سنٹرڈ ہو چکا ہے۔
- CTRL+J کو پریس کریں۔ ٹیکسٹ اب جسٹیفائیڈ ہو چکا ہے۔

آئیکن استعمال کرتے ہوئے جسٹیفائی اور سینٹر کرنا۔

- ٹیکسٹ کو ہائی لائٹ کریں۔
- سینٹر کے آئیکن  کو جو کہ سٹینڈرڈ ٹول بار پر ہے، کلک کریں۔ ٹیکسٹ اب سنٹرڈ ہو چکا ہے۔
- جسٹیفائیڈ کے آئیکن  کو کلک کریں۔ ٹیکسٹ اب جسٹیفائیڈ ہو چکا ہے۔

یہ تینوں طریقے اختیار کر کے دیگر پیراگراف الائنمنٹس بھی کی جا سکتی ہیں۔

ہینگنگ انڈینٹ (Hanging Indent)

ہینگنگ انڈینٹ فیچر ہر لائن کو انڈینٹ کرتا ہے سوائے پہلی لائن کے اور اس کی مقدار By field میں بتائی جاتی ہے جیسا کہ مندرجہ ذیل مثال میں دکھایا گیا ہے۔

Hanging Indent: The Hanging Indent feature indents the first line of the paragraph from margin by the amount specified in the Left field. The amount in the Left field plus the amount specified in the By field indent all subsequent lines.

جب آپ مندرجہ ذیل پیراگراف ٹائپ کرنا شروع کرتے ہیں تو آپ دیکھتے ہیں کہ آپ کا پیراگراف ایک انچ دونوں طرف سے انڈینٹ ہو گیا ہے۔ مائیکروسوفٹ ورڈ میں جب آپ ایک نیا پیراگراف ٹائپ کرتے ہیں تو جو سٹینگز پچھلے پیراگراف کی تھیں وہی نئے پیراگراف پر اپلائی ہو جاتی ہیں۔ اگر آپ چاہتے ہیں تو آپ انڈینٹ کو ری سیٹ کر سکتے ہیں۔ مندرجہ ذیل ٹائپ کریں:

Hanging Indent: The hanging indent feature indents the first line by the amount specified in the Left field. Subsequent lines are indented by the amount specified in the Left field plus the amount specified in the By field.

- اس ٹائپ شدہ پیراگراف کو ہائی لائٹ کریں۔
- مینیو میں سے Format>Paragraph کی کمانڈ منتخب کریں۔
- انڈینٹس اور سپیسنگ ٹیب کو منتخب کریں۔
- Special Field میں پل ڈاؤن مینیو کو کلک کر کے اوپن کریں۔
- ہینگنگ (Hanging) کو کلک کریں۔
- By box میں "2.0" ٹائپ کریں۔
- OK کو کلک کریں۔
- "Hanging Indent" کے بعد آنے والے کولن پر کرسر کو رکھیں۔
- ٹیب کی (Tab key) کو پریس کریں۔

فونٹس (Fonts)

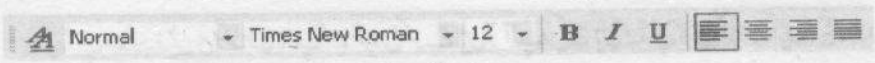
مائیکروسوفٹ ورڈ میں ٹیکسٹ شکل (typeface) بتانے کے لیے فونٹ کی اصطلاح استعمال ہوتی ہے۔ آپ فونٹ (اپنے ٹیکسٹ کے حروف کی شکل اور بناوٹ کی ٹائپ) تبدیل کر سکتے ہیں۔ اس فیچر (Feature) کا تفصیلی ذکر درج ذیل ہے:

مینیو کو استعمال کرتے ہوئے فونٹ تبدیل کرنا۔

- ٹائپ کریں۔ Arial Courier, Times New Roman
- Arial کو ہائی لائٹ کریں۔
- مینیو سے Format>Font منتخب کریں۔
- فونٹ کے ٹیب کو چوز کریں۔
- فونٹ فیلڈ کے نیچے والے باکس میں "Arial" کو کلک کریں۔
- OK کو کلک کریں۔
- "Courier" کو ہائی لائٹ کریں۔
- مینیو سے Format>Font کی کمانڈ منتخب کریں۔
- فونٹ کے ٹیب کو منتخب کریں۔
- فونٹ فیلڈ کے نیچے والے باکس میں "Courier New" کو کلک کریں۔
- OK کو کلک کریں۔
- "Times New Roman" کو ہائی لائٹ کریں۔
- فونٹ کے ٹیب کو منتخب کریں۔
- فونٹ فیلڈ کے نیچے والے باکس میں سے "Times New Roman" کو کلک کریں۔
- OK کو کلک کریں۔
- آپ کا ٹیکسٹ اب کچھ اس طرح نظر آئے گا

"Arial Courier Times New Roman"

فارمیٹنگ ٹول بار کے استعمال سے فونٹ کو تبدیل کرنا

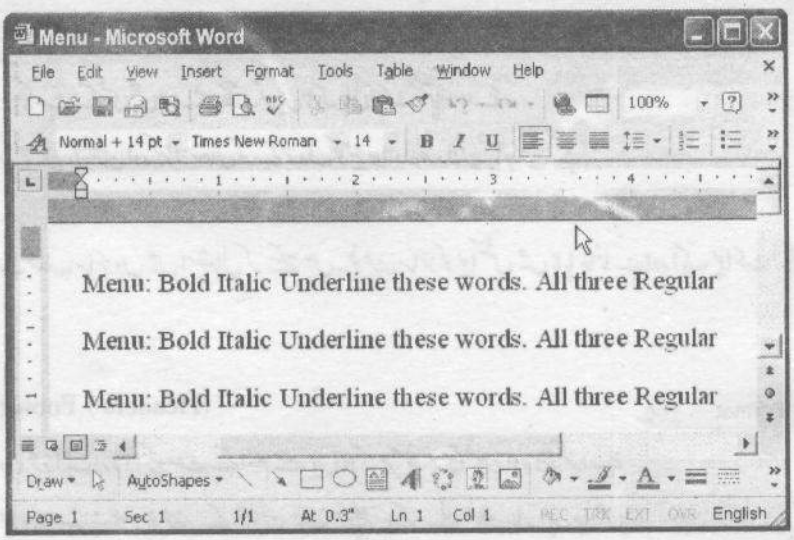


- "Arial Courier Times New Roman" کو ہائی لائٹ کریں۔
- CTRL+Spacebar کو پریس کریں۔ CTRL+Spacebar فارمیٹنگ کو اصلی حالت میں سیٹ کر دیتا ہے۔
- "Arial" کو ہائی لائٹ کریں۔
- فارمیٹنگ ٹول بار پر فونٹ پل ڈاؤن مینیو کو کلک کر کے اوپن کریں۔
- "Arial" کو کلک کریں۔
- اَب "Courier" کو ہائی لائٹ کریں۔
- فارمیٹنگ ٹول بار پر فونٹ پل ڈاؤن مینیو کو کلک کر کے اوپن کریں۔
- Courier کو کلک کریں۔
- اب دوبارہ "Times New Roman" کو ہائی لائٹ کریں۔
- فارمیٹنگ ٹول بار پر فونٹ پل مینیو کو کلک کر کے اوپن کریں۔
- "Times New Roman" کو کلک کریں۔
- آپ کا ٹیکسٹ اس شکل میں نظر آنا چاہیے

"Arial Courier Times New Roman"

بولڈ، انڈر لائن اور ایٹالک (Bold, Underline, and Italic)

جب آپ ورڈ کو استعمال کر رہے ہوں تو آپ اپنے ٹیکسٹ کو بولڈ، انڈر لائن (Underline) اور ایٹالک (Italic) کر سکتے ہیں۔ آپ ان فیچرز کو ملا بھی سکتے ہیں۔ دوسرے لفظوں میں یوں کہہ لیں کہ آپ ٹیکسٹ کے سنگل ٹکڑے کو بولڈ، انڈر لائن اور ایٹالک کر سکتے ہیں۔ جو مشق مندرجہ ذیل میں آ رہی ہے، اس میں آپ ورڈ پروگرام میں بولڈ، انڈر لائن اور ایٹالک کرنے کے مختلف طریقے سیکھیں گے۔ آپ مینیو، آئیکن یا کیز کی مدد سے بولڈ، انڈر لائن اور ایٹالک کرنا سیکھیں گے۔



مینیو، آئیکن اور کیز کے استعمال سے بولڈ کرنا B

- اُس لائن پر جو کہ "Menu" سے شروع ہوتی ہے، "Bold" کے لفظ کو ہائی لائٹ کریں۔ ایسا کرنے کے لیے کرسر کو لفظ "Bold" کے حرف "B" سے پہلے رکھیں۔ F8 کی کو پریس کریں۔ پھر رائٹ ایرو کو پریس کریں جب تک کہ پورا لفظ ہائی لائٹ نہ ہو جائے۔
- مینیو سے Format>Font کو منتخب کریں۔ فونٹ کا ڈائیلاگ باکس اوپن ہو جاتا ہے۔
- فونٹ سٹائل باکس میں Bold کو کلک کریں۔
- OK کو کلک کریں تا کہ ڈائیلاگ باکس بند ہو جائے۔
- ٹیکسٹ کے کسی بھی حصہ میں کلک کریں تا کہ ہائی لائٹ ختم ہو جائے۔ اب وہ لفظ بولڈ ہو چکا ہے۔

نوٹ:۔ آپ اپنی سلیکشن کا اثر پری ویو ونڈو میں دیکھ سکتے ہیں۔ بولڈ فارمیٹ کو ٹرن آف کرنے کے لیے ریگولر کے آپشن پر کلک کریں۔

- اُس لائن پر جو کہ لفظ "Icon" سے شروع ہوتی ہے، "Bold" کے لفظ کو ہائی لائٹ کریں۔ ایسا کرنے کے لیے کرسر کو لفظ "Bold" کے حرف "B" سے پہلے رکھیں۔ F8 کی کو پریس کریں۔ پھر رائٹ ایرو کو پریس کریں جب تک کہ پورا لفظ ہائی لائٹ نہ ہو جائے۔
- ٹول بار سے آئیکن پر کلک کریں۔

نوٹ:۔ بولڈ کے اثر کو ختم کرنے کے لیے ٹیکسٹ کو ہائی لائٹ کریں اور دوبارہ بولڈ کے آئیکن پر کلک کریں۔

- ٹیکسٹ کے کسی بھی حصہ میں کلک کریں تا کہ ہائی لائٹ ختم ہو جائے۔
- اُس لائن پر جو کہ لفظ "Keys" سے شروع ہوتی ہے، "Bold" کے لفظ کو ہائی لائٹ کریں۔ ایسا کرنے کے لیے کرسر کو لفظ "Bold" کے حرف "B" سے پہلے رکھیں۔ F8 کی پریس کریں۔ پھر رائٹ ایرو کو پریس کریں جب تک کہ پورا لفظ ہائی لائٹ نہ ہو جائے۔
- Ctrl+b دبائیں (حرف b دباتے وقت Ctrl key دبائے رکھیں)۔

نوٹ:۔ بولڈ فارمیٹنگ ختم کرنے کے لیے دوبارہ Ctrl+b دبائیں۔ آپ spacebar+Ctrl دبا کر بھی فارمیٹنگ ختم کر سکتے ہیں۔

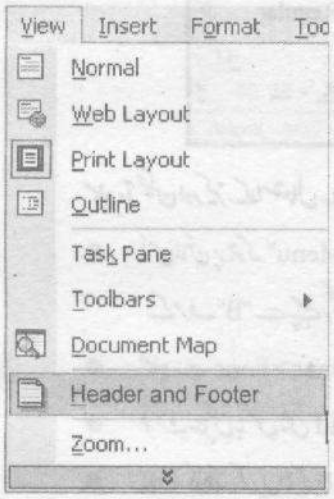
- ٹیکسٹ کے کسی بھی حصہ میں کلک کریں تا کہ ہائی لائٹ ختم ہو جائے۔

اسی طرح مینیو، آئیکن اور کیز استعمال کر کے حروف کو ٹیڑھا اور انڈر لائن کیا جا سکتا ہے۔

فونٹ سائز:

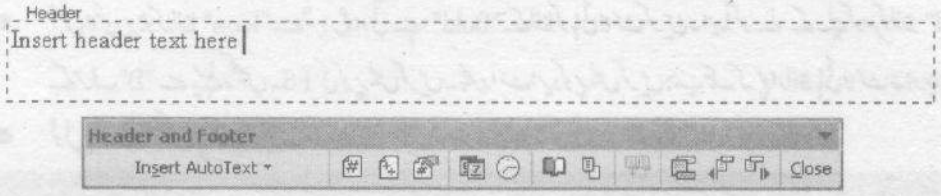
ہر فونٹ مختلف سائزوں میں استعمال کر سکتے ہیں۔ فونٹ سائز کو پوائنٹس میں ناپا جاتا ہے اور ایک پوائنٹ انچ کا $\frac{1}{72}$ واں حصہ ہوتا ہے۔

ہیڈرز/ فوٹرز (Headers / Footers)



ہیڈرز اور فوٹرز ایک ورڈ ڈاکیومنٹ کے اہم حصے ہیں جو کہ ہر صفحہ پر پیج نمبرز اور ہیڈنگز دینے کے کام آتے ہیں۔ ہیڈرز اور فوٹرز کے آپشن تک پہنچنے کے لیے مینیو بار پر جائیں اور View>>Header and Footer سلیکٹ کریں۔

ایک ڈوٹڈ (Dotted) لائن کا باکس خود بخود ظاہر ہو جائے گا جسے "Header" کہتے ہیں۔ اس کے ساتھ ساتھ ایک سب مینیو ہوتا ہے جس کی مدد سے ہیڈر اور فوٹر کی ٹیز کو ایڈجسٹ کیا جا سکتا ہے۔ کرسر پہلے سے ہی ہیڈر باکس میں موجود ہوتا ہے۔ اگر آپ موجودہ پیج کے نیچے جائیں جو اس وقت ورڈ میں کھلا ہوا ہے تو آپ کو ڈاٹڈ لائن کا باکس نظر آئے گا جسے فوٹر کہتے ہیں۔ ہیڈر یا فوٹر میں ٹیکسٹ کو ایڈٹ کرنے کے لیے کسی بھی باکس کے اندر کرسر کو کلک کریں اور ٹائپ کرنا شروع کر دیں۔ اپنے ڈاکیومنٹ میں پیج نمبرز کا اضافہ کرنے کے لیے فوٹر کے اندر کرسر کو کلک کریں اور پھر اُسی آئیکن کو کلک کریں جو ایک کاغذ کے سیٹ کی طرح کا ہے اور اُس کے اندر "#" بنا ہوا ہے۔ وہاں پر پیج نمبر لکھا جائے گا اور آپ کے ڈاکیومنٹ کے تمام صفحات پر نظر آئے گا۔



بُلٹڈ اور نمبرڈ لسٹ (Bulleted and Numbered List)

- Bulleted List کے آئیکن کو یا Numbered List کے آئیکن کو جو کہ فارمیٹنگ ٹول بار پر ہے کلک کریں۔
- پہلی اینٹری ٹائپ کریں اور ENTER کو پریس کریں۔ یہ اگلی لائن پر ایک نیا نمبر یا بُلٹ (Bullet) بنائے گی۔ اگر آپ ایک نئی لائن شارٹ کرنا چاہتے ہیں جس میں بُلٹ یا نمبر نہ ہو تو SHIFT کی کو ہولڈ (Hold) کرتے ہوئے ENTER پریس کریں۔
- اینٹریز کو ٹائپ کرتے رہیں اور جب آپ ٹائپنگ ختم کر دیں تو دو دفعہ ENTER پریس کریں تا کہ لسٹ ختم ہو جائے۔

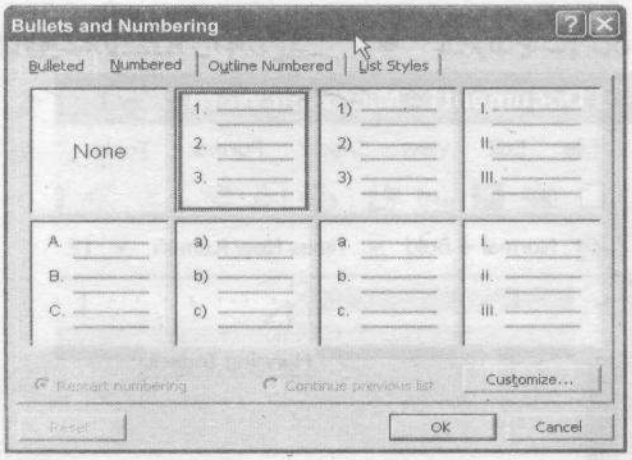
نوٹ:۔ آپ پہلے ٹیکسٹ ٹائپ کر سکتے ہیں، مطلوبہ حصے کو ہائی لائٹ کر کے بلٹس یا نمبرز دینے کے لیے Bulleted List یا Numbered List پر کلک کر سکتے ہیں۔

ملٹی لیول فہرستیں بنانے کے لیے فامیٹنگ ٹول بار پر Increase Indent اور Decrease Indent کے بٹن استعمال کریں۔ Bullets and Numbering ڈائیلاگ باکس کو استعمال کرتے ہوئے آپ بُلٹ امیج اور نمبرنگ (Numbering) فارمیٹ تبدیل کر سکتے ہیں۔

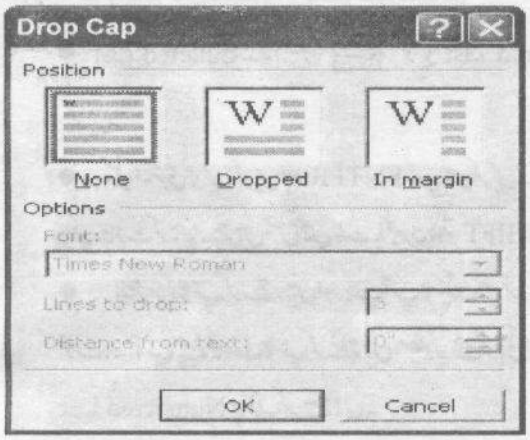
- ساری لسٹ کو ہائی لائٹ کریں تا کہ بُلٹس یا نمبرز کو تبدیل کیا جا سکے یا لسٹ کے اندر کسی ایک لائن پر کرسر کو رکھیں تا کہ ایک سنگل بُلٹ کو تبدیل کیا جا سکے۔
- Format/Bullets and Numbering کو مینیو بار میں سے سلیکٹ کر کے اُس کا ڈائیلاگ باکس کھولیں یا لسٹ کے درمیان رائٹ کلک کریں اور شارٹ کٹ مینیو سے Bullets and Numbering کو سلیکٹ کریں۔
- دی گئی سات چوائسز (Choices) میں سے لسٹ سٹائل کو سلیکٹ کریں یا Picture بٹن کو کلک کریں تا کہ ایک مختلف آئیکن سلیکٹ کر سکیں۔

نمبرڈ لسٹ سٹائل کو منتخب کرنے کے لیے Numbered ٹیب کو کلک کریں۔

- جب ختم کر لیں تو OK کو کلک کریں۔



ڈراپ کیپس (Drop Caps)



ایک ڈراپ کیپ ایک بڑا حرف ہے جس سے پیرا گراف شروع ہوتا ہے اور اس سے کافی زیادہ لائنز ڈراپ ہو جاتی ہیں جیسا کہ مندرجہ ذیل میں دکھایا گیا ہے۔

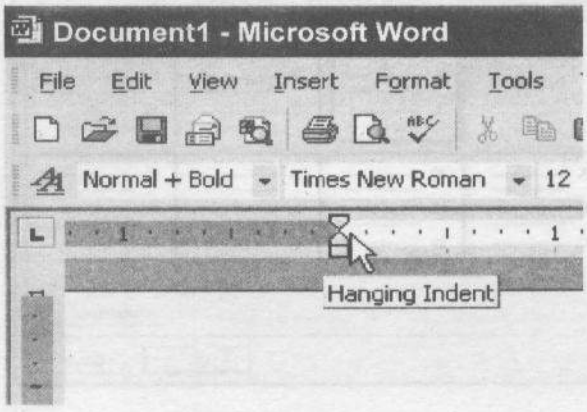
آپ ایک پیرا گراف میں مندرجہ ذیل سٹیپس (Steps) کی مدد سے ڈراپ کیپ کر سکتے ہیں:

- اُس پیرا گراف کے اندر کرسر کو رکھیں جس کا پہلا حرف ڈراپ کرنا ہو۔
- مینیو بار میں سے Format/Drop Caps کی کمانڈ سلیکٹ کریں۔
- ڈراپ کیپ کا ڈائیلاگ باکس آپ کو اجازت دیتا ہے کہ ڈراپ کیپ کی پوزیشن، اُس کا فونٹ، کتنی لائنز تک ڈراپ کرنا ہے اور باڈی ٹیکسٹ سے فاصلہ کو سلیکٹ کر سکیں۔
- جب تمام سلیکشنز ہو جائیں تو OK کو کلک کریں۔
- ایک ڈراپ کیپ کو تبدیل کرنے کے لیے Format/Drop Cap کو دوبارہ سلیکٹ کریں تا کہ ایٹریبیوٹس کو تبدیل کیا جا سکے یا لفظ پر کلک کریں اور لفظ کو حرکت دینے اور اس کا سائز تبدیل کرنے کے لیے ہینڈلز کو استعمال کریں۔

پیج مارجنز (Page Margins)

ایک ڈاکیومنٹ کے پیج مارجن کو تبدیل کرنے کے لیے پیج کے رولرز کو اور Page Setup ونڈو کو استعمال کیا جا سکتا ہے۔ رولر کا طریقہ پہلے بیان کیا جاتا ہے۔

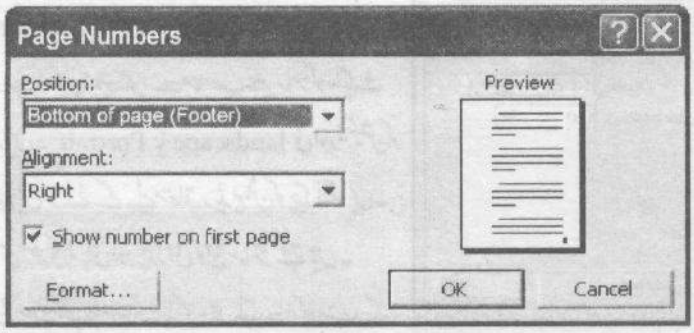
- کرسر کو اُسی جگہ حرکت دیں جس جگہ پر سفید رولر گرے ہونا شروع ہو جاتا ہے۔
- جب کرسر ڈبل اینڈڈ ایرو بن جاتا ہے تو ماؤس سے کلک کریں اور مارجن انڈیکیٹر کو اپنی سلیکٹ کردہ جگہ پر ڈریگ کر کے لے جائیں۔
- جب مارجن سیٹ ہو جائے تو ماؤس کو چھوڑ دیں۔
- مارجنز کو ڈائیلاگ باکس کی مدد سے بھی تبدیل کیا جا سکتا ہے۔



پیج نمبرز کو انسرٹ کرنا (Insert Page Numbering)

جیسا کہ پہلے نوٹ کیا جا چکا ہے، آپ ایک ڈاکیومنٹ میں پیج نمبرز داخل کر سکتے ہیں اور ایسا کرنے کے لیے ہیڈر اور فوٹر کی پروپرٹیز کو استعمال کر سکتے ہیں۔ آپ ہیڈر/فوٹر پروپرٹیز استعمال کیے بغیر بھی پیج نمبر دے سکتے ہیں۔

مینیو بار پر ''انسرٹ'' کو کلک کریں اور پھر ''پیج نمبرز۔۔۔'' کو کلک کریں۔ ڈاکیومنٹ کے ٹاپ پر ''پیج نمبرز'' کا ڈائیلاگ باکس ظاہر ہو جاتا ہے۔



آپ ''پوزیشن'' ڈراپ باکس استعمال کر کے پیج کے اوپر یا نیچے پیج نمبر کی پوزیشن کو سیٹ کر سکتے ہیں۔ آپ ''الائنمنٹ'' کے ڈراپ باکس کو بھی استعمال کر سکتے ہیں۔ اس کو پیج نمبر کی الائنمنٹ کے لیے استعمال کیا جاتا ہے اور یہ الائنمنٹ بائیں، دائیں، درمیان، اندر یا ڈاکیومنٹ کے باہر ہو سکتی ہے۔ آپ ''شو نمبر اون فرسٹ پیج'' کے چیک باکس کو کلک کر کے چیک مارک لگا سکتے ہیں تا کہ پہلے پیج پر نمبر ظاہر ہو اور اسے کلک کر کے چیک مارک ختم کر سکتے ہیں تا کہ پہلے پیج پر نمبر نہ آئے۔ پیج نمبر کو دائیں طرف الائن کرنے کے لیے ''الائنمنٹ'' ڈراپ باکس استعمال کریں۔ پھر OK پر کلک کریں۔ دیکھیں پیج نمبر پیج کے دائیں طرف الائن ہو گیا ہو گا۔

نان پرنٹنگ کریکٹرز (Non-Printing Characters) ¶

ہم سکرین پر خاص علامتیں دیکھ سکتے ہیں جن سے علم ہوتا ہے کہ آپ نے "Enter" کہاں پریس کیا ہے، کس جگہ "Spacebar" پریس کیا ہے یا اپنے ڈاکیومنٹ میں کس جگہ "Tab" کو استعمال کیا ہے۔ یہ نہ پرنٹ ہونے والے کریکٹرز ڈاکیومنٹس کی ایڈیٹنگ میں مددگار ثابت ہوتے ہیں اور اس بات کو یقینی بناتے ہیں کہ ٹیکسٹ اُسی جگہ پر ہے جس جگہ پر آپ اسے چاہتے ہیں۔ ان خاص کریکٹرز کو ظاہر کرنے کے لیے سٹینڈرڈ ٹول بار پر Show/Hide کا بٹن جو کہ سٹینڈرڈ ٹول بار پر ہوتا ہے کو کلک کریں۔

ڈاکیومنٹ پری ویو اور پرنٹنگ (Preview and Printing a Document)

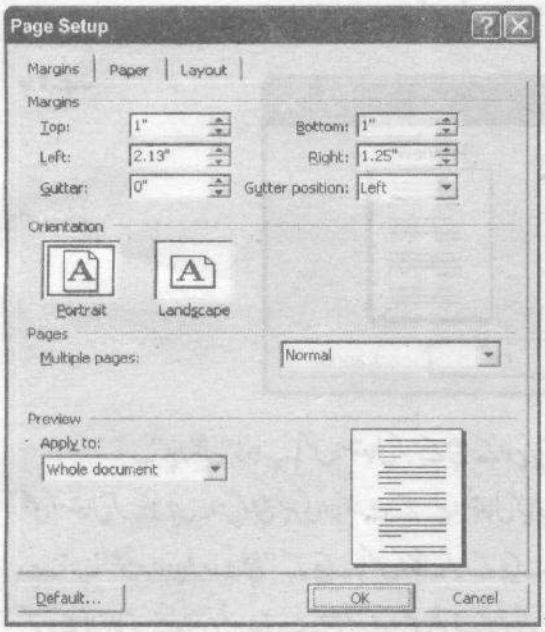
آپ ایک ڈاکیومنٹ کو پرنٹر پر بھیجنے سے پہلے پری ویو کر سکتے ہیں۔ ایسا کرنے کے لیے مندرجہ ذیل سٹیپس کریں۔

- سٹینڈرڈ ٹول بار میں سے "Print Preview" کو کلک کریں۔ MS ورڈ پریویو (Preview) موڈ میں ایک نئی ونڈو کھول دیتا ہے۔ آپ کو خود بخود پتہ چل جاتا ہے کہ آپ پرنٹ پریویو میں ہیں کیونکہ ٹائٹل بار میں "Preview" کا لفظ ڈاکیومنٹ کے نام کے بعد لکھا ہوتا ہے۔ یہ بھی نوٹ کریں کہ سارے کا سارا ڈاکیومنٹ ونڈو میں پورا فٹ (Fit) آتا ہے۔
- پرنٹ پری ویو کے ٹول بار میں "Close" کے بٹن کو کلک کریں تا کہ پرنٹ پری ویو موڈ سے نکل کر ڈاکیومنٹ کی طرف واپس آ جائیں۔

- فارمیٹنگ ٹول بار پر "Print" کے  کو کلک کریں تا کہ ڈاکیومنٹ پرنٹ ہو جائے۔

پیج سائز اور اورینٹیشن (Page Size and Orientation)

پیج Setup کے ڈائیلاگ باکس کے اندر آپ صفحہ کی اورینٹیشن کو تبدیل اور کنٹرول کر سکتے ہیں۔



- File/Page Setup کو سلیکٹ کریں اور Paper Size کے ٹیب کو منتخب کریں۔
- ڈراپ ڈاؤن مینیو میں سے مناسب پیپر سائز کو سلیکٹ کریں۔ Portrait یا landscape کی اورینٹیشن کو تبدیل کرنے کے لیے متعلقہ ریڈیو بٹن کو چیک کریں۔
- کسی صفحہ کی لمبائی اور چوڑائی کو پیج سائز کہتے ہیں۔ پیج سیٹ اَپ ڈائیلاگ باکس میں ایک ڈاکیومنٹ کے پیج کا سائز لکھا ہوتا ہے۔
- فائل مینیو کو کلک کریں اور پیج سیٹ اَپ کو سلیکٹ کریں۔
- پیپر سائز کے ٹیب کو سلیکٹ کریں اور دیے گئے سٹینڈرڈ سائزز (Sizes) میں سے مناسب پیج سائز کو سلیکٹ کریں۔

کچھ سٹینڈرڈ سائزز اور اُن کی ڈائمینشنز (Dimensions) مندرجہ ذیل دی گئی ہیں۔

Letter: paper size of 8.5by 11 inches
Legal: paper size of 8.5 by 14 inches
A4: paper size of 8.27 by 10.5 inches

7.6 خودمختار فیچرز (Automatic Features)

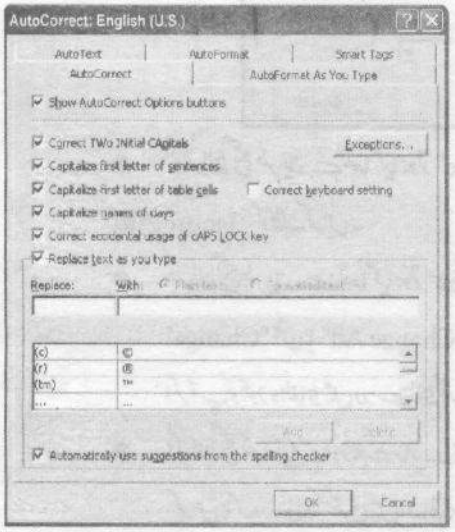
جب آپ ایک ڈاکیومنٹ میں ٹیکسٹ ٹائپ کرتے ہیں تو سُرخ یا سبز لہراتی ہوئی لائنز بعض ورڈز کے نیچے ظاہر ہو جاتی ہیں۔ یہ نان پرنٹنگ لائنز Automatic Spell Check کی وجہ سے آتی ہیں اور ظاہر کرتی ہیں کہ یہ لفظ ورڈ ڈکشنری میں نہیں پہچانا گیا۔ سبز لہراتی لائنز یہ ظاہر کرتی ہیں کہ یہ جملہ گرائمر کے لحاظ سے صحیح نہیں ہے۔ مندرجہ ذیل مثال میں، لفظ "example" کے spelling صحیح طرح سے نہیں لکھے گئے اور ایسا سُرخ لہراتی لائن جو کہ اس کے نیچے ہے، سے ظاہر ہوتا ہے۔

This is an example of Automatic Spell Check

"Auto Correct" ورڈ کا ایک خودمختار ایڈیٹنگ فیچر ہے آٹو کوریکٹ، خود بخود غلط لکھے گئے ٹیکسٹ کو صحیح کرتا ہے۔ مثال کے طور "Teh" کو "The" سے تبدیل کر دیا جاتا ہے اور یہ اُس وقت ہوتا ہے جب آپ سپیس بار کو پریس کرتے ہیں۔ ورڈ میں کافی تعداد میں پہلے سے موجود Auto Correct اینٹریز ہوتی ہیں۔ اور یوزر اینٹریز کا اضافہ کر سکتا ہے۔

آٹوٹیکسٹ (Auto Text)

"Auto Text" ورڈ کا ایک اور آٹومیٹک فیچر ہے۔ آٹوٹیکسٹ کی اینٹریز (Enteries) ایک ڈاکیومنٹ کو جلدی اسمبل (Assemble) کرنے میں استعمال ہوسکتی ہیں۔ جیسے ہی آپ ایک عام طور سے استعمال ہونے والا لفظ ٹائپ کرنا شروع کرتے ہیں، ایک پیلے رنگ کا آٹوٹیکسٹ فلیگ (Flag) ظاہر ہوجائے گا۔ فلیگ کو قبول کرنے کے لیے، اینٹر کی کو پریس کریں۔ آٹوٹیکسٹ اینٹری کو نظر انداز کرنے کے لیے ٹائپنگ جاری رکھیں، آٹو فلیگ غائب ہوجائے گا۔



آٹوکمپلیٹ (Auto Complete)

آٹوکمپلیٹ آپ کو اجازت دیتا ہے کہ مخصوص شناختی کریکٹرز ٹائپ کرکے پوری آئٹمز کو انسرٹ کیا جا سکے۔ جیسا کہ تاریخ اور آٹوٹیکسٹ اینٹریز۔ مندرجہ ذیل مثال میں "Date" ایک Auto Complete اینٹری بن گئی ہے۔

ورڈ خود بخود آٹوکوریکٹ فیچر کی وجہ سے عام طور سے غلط لکھے گئے الفاظ اور پنکچوایشن (Punctuation) مارکس صحیح کردیتا ہے۔ اُن تمام خود بخود صحیح ہوجانے والے الفاظ کی لسٹ دیکھنے کے لیے Tools/Auto Correct کو سلیکٹ کریں۔ اگر ٹولز مینیو میں یہ کمانڈ نظر نہ آئے تو اس کے نیچے بنے ہوئے ڈبل ایروز کے نشان پر کلک کریں۔ اس سے ٹولز مینیو کی تمام کمانڈز ظاہر ہوجائیں گی۔ ان میں سے آپ آٹوکوریکٹ کی کمانڈ منتخب کرسکتے ہیں۔

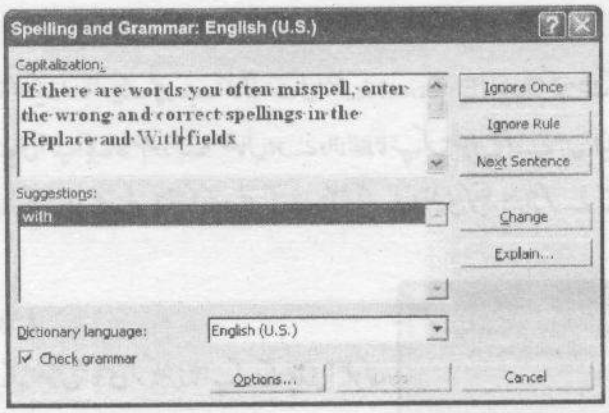
اس میں کافی زیادہ اوپشنز میں شامل ہیں جیسا کہ ایک لفظ کے پہلے دو حروف کا غلطی سے کیپٹالائز (Capatalize) ہونا اور ایک فقرے کے پہلے حرف کو خود بخود بڑا لکھنا۔ اگر ایسے الفاظ ہیں جن کو آپ اکثر غلط ٹائپ کردیتے ہیں تو "Replace" اور "With" فیلڈز میں غلط اور صحیح سپیلنگز اینٹر کریں۔

سپیلنگ اور گرائمر چیک (Spelling and Grammar Check)

اگر سپیلنگ اور گرائمر چیکنگ کا آپشن آن ہے تو ٹائپنگ کے دوران ورڈ خود بخود سپیلنگ اور گرائمر کی ایررز کو چیک کرتا ہے۔ ڈاکیومنٹ میں سپیلنگ کی ایررز ایک سُرخ انڈر لائن سے ظاہر ہوتی ہیں۔ سبز انڈر لائن گرائمر کی ایررز کو ظاہر کرتی ہے۔ اس فیچر کو ڈسِ ایبل کرنے کے لیے مینیو بار میں سے Tools/Options کو سلیکٹ کریں اور Spelling and Grammar Tab کو سلیکٹ کریں۔ "Check spelling as you type" اور "Check Grammar as you type" کو اَن چیک کریں اور OK پر کلک کریں۔

سپیلنگ اور گرائمر چیکر (Checker) کو استعمال کرنے کے لیے مندرجہ ذیل سٹیپس کو فالو (Follow) کریں۔

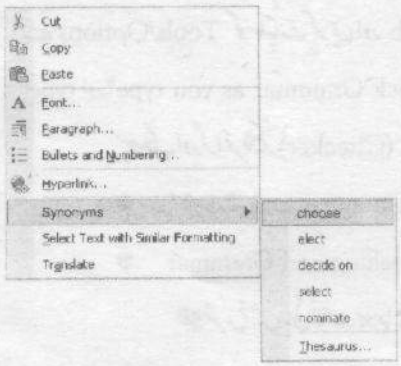
- مینیو بار میں سے Tools|Spelling and Grammar کو سلیکٹ کریں۔
- Spelling and Grammar ڈائیلاگ باکس آپ کے ڈاکیومنٹ میں پہلی ایرر کی نشان دہی کردے گا اور غلط لکھا گیا لفظ سُرخ میں ہائی لائٹ ہوجائے گا۔



- اگر لفظ صحیح طریقہ سے لکھا گیا ہو تو "Ignore" کے بٹن کو کلک کریں یا اگر یہ لفظ ڈاکیومنٹ میں ایک سے زیادہ جگہ پر آ رہا ہو تو "Ignore All" کو کلک کریں۔
- اگر لفظ صحیح طریقہ سے نہ لکھا گیا ہو تو سپیلنگ میں دیے گئے "Suggestions" باکس میں سے درست لفظ منتخب کریں اور "Change" بٹن یا "Change All" کے بٹن کو کلک کریں تاکہ ڈاکیومنٹ میں جتنی جگہ بھی وہ لفظ آ رہا ہو وہ صحیح ہو جائے۔
اگر آپ کو مجوزہ الفاظ میں درست لفظ نظر نہ آئے تو Not in Dictionary میں صحیح سپیلنگ لکھ کر Change کو کلک کریں۔
- اگر لفظ صحیح لکھا گیا ہے اور بہت سارے ڈاکیومنٹس میں آ رہا ہو جو آپ استعمال کرتے ہیں جیسا کہ آپ کا نام تو "Add" بٹن کو کلک کریں تاکہ ورڈ کی ڈکشنری میں اضافہ ہو جائے اور ایسا لکھا ہوا لفظ دوبارہ غلط تصور نہ ہوگا۔

جب تک کہ Spelling and Grammar ڈائیلاگ باکس میں Check Grammar باکس چیکڈ ہو، ورڈ، ڈاکیومنٹ کی گرائمر بھی چیک کرے گا اور سپیلنگ بھی۔ اگر آپ نہیں چاہتے کہ گرائمر چیک ہو تو اس باکس میں سے چیک مارک کو مٹا دیں۔ اس کے علاوہ مندرجہ ذیل سٹیپس کو فالو کریں تاکہ گرائمر کو صحیح کیا جا سکے۔

- اگر ورڈ کوئی گرائمر کی غلطی ڈھونڈتا ہے، تو یہ سپیلنگ ایررز کے باکس میں دکھائی دیں گی۔ غلطی سبز ٹیکسٹ سے ہائی لائٹ ہوگی۔
- Suggestions باکس میں بہت ساری تجویزیں (Suggestions) دی جا سکتی ہیں۔ وہ کوریکشن (Correction) سلیکٹ کریں جو کہ زیادہ قریب تر ہو اور Change کو کلک کریں۔
- اگر تبدیلی کی ضرورت نہ ہو تو Ignore کے بٹن کو کلک کریں۔

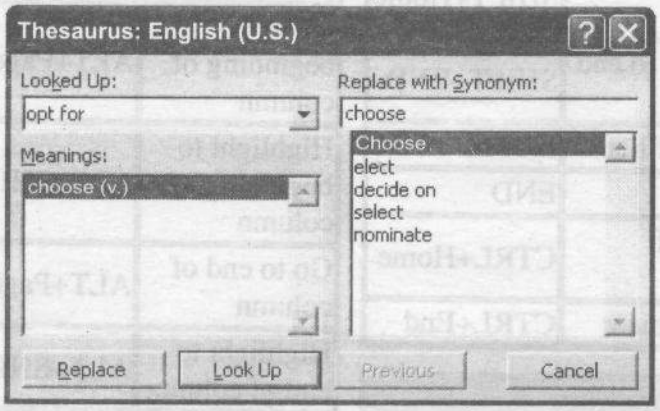


سائنونیمز (Synonyms)

Synonyms ورڈ میں ایک نیا فیچر ہے جس کی مدد سے ہم معنی الفاظ / سائنونیمز ڈھونڈے جا سکتے ہیں۔ لفظ پر صرف رائٹ کلک کریں اور شارٹ کٹ مینیو میں سے Synonyms کو سلیکٹ کریں۔ تجویز کی گئی لسٹ میں سے وہ لفظ ہائی لائٹ کریں جو آپ استعمال کرنا چاہتے ہیں یا زیادہ اوپشنز کے لیے.... Thesaurus کو کلک کریں۔

دیسارس (Thesaurus)

اگر آپ کسی لفظ کو بار بار استعمال کرنے سے گریز کرنا چاہتے ہیں اور چاہتے ہیں کہ اس کے لیے ایک اچھا متبادل لفظ مل جائے تو دیسارس کو استعمال کریں۔ دیسارس کو استعمال کرنے کے لیے مینیو بار میں سے Tools/Language/Thesaurus کو سلیکٹ کریں یا جیسا کہ اوپر بیان کیا گیا ہے اس کو Synonyms کے شارٹ کٹ مینیو میں سے سلیکٹ کریں۔



ونڈوز میں معانی اور مترادف الفاظ کی لسٹ دی گئی ہے۔ الفاظ جو کہ Meanings باکس میں ہوں، ان کو ڈبل کلک کریں یا Look Up بٹن کو کلک کریں تا کہ ایک جیسے الفاظ دیکھ سکیں۔ Replace with Synonyms کے باکس میں الفاظ کو ڈبل کلک کریں تا کہ اُن الفاظ کے سائنونیمز دیکھ سکیں۔ اُس لفظ کو ہائی لائٹ کریں جسے آپ استعمال کرنا چاہتے ہیں اور Replace بٹن کو کلک کریں۔

نوٹ:۔ کسی کمانڈ میں جمع کی علامت (+) ظاہر کرتی ہے کہ کیز بیک وقت دبائیں۔

Action	Keystroke
Document actions	
Open a file	CTRL+O
New file	CTRL+N
Close a file	CTRL+W
Save As	F12
Save	CTRL+S or SHIFT+F12
Print Preview	CTRL+F2
Print	CTRL+P
Show/Hide paragraph symbols	CTRL+*
Spelling and grammar	F7
Help	F1
Find	CTRL+F
Replace	CTRL+H
Go To	CTRL+G

Action	Keystroke
Text Style	
Font face	CTRL+SHIFT+F
Font size	CTRL+SHIFT+P
Bold	CTRL+B
Italics	CTRL+I
Underline	CTRL+U
Double underline	CTRL+SHIFT+D
Word underline	CTRL+SHIFT+W
All caps	CTRL+SHIFT+A
Change case	SHIFT+F3
Subscript	CTRL+=
Superscript	CTRL+SHIFT+=
Make web hyperlink	CTRL+K

Cursor movement	
Select all - entire document	CTRL+A
Select from cursor to beginning of line	SHIFT+Home
Select from cursor to end of line	SHIFT+END
Go to beginning of line	HOME
Go to end of line	END
Go to beginning of document	CTRL+Home
Go to end of document	CTRL+End

Formatting	
Cut	CTRL+X
Copy	CTRL+C
Paste	CTRL+V
Undo	CTRL+Z
Redo	CTRL+Y
Format painter	CTRL+SHIFT+C
Left alignment	CTRL+L
Center alignment	CTRL+E
Right alignment	CTRL+R
Justified	CTRL+J
Delete previous word	CTRL+Backspace
Apply bulleted list	CTRL+SHIFT+L
Indent	CTRL+M
Page break	CTRL+Enter

Tables	
Go to next cell	Tab
Go to previous cell	SHIFT+Tab
Go to beginning of column	ALT+PageUp
Highlight to beginning of column	ALT+SHIFT+PageUp
Go to end of column	ALT+PageDown
Highlight to end of column	ALT+SHIFT+PageDown
Go to beginning of row	ALT+Home
Highlight to	ALT+SHIFT+Home
beginning of row	
Go to end of row	ALT+End
Highlight to end of row	ALT+SHIFT+End
Column break	CTRL+SHIFT+Enter

Miscellaneous	
Copyright symbol - ©	ALT+CTRL+C
Date field	ALT+SHIFT+D
Go to footnotes	ALT+CTRL+F
Show/Hide ¶	CTRL+SHIFT+8
Thesaurus	SHIFT+F7

مشق

1۔ خالی جگہ پُر کریں۔

(i) 8.5x11 انچ کے پیپر سائز کو _____ سائز کہتے ہیں۔

(ii) فونٹ کو _____ بھی کہتے ہیں۔

(iii) MS ورڈ میں ایک ڈاکیومنٹ کو سیو کرنے کا کی بورڈ شارٹ کٹ ہے _____

(iv) MS ورڈ کی مینیو بار میں _____ آئٹمز ہیں۔

(v) سٹارٹ بٹن _____ بار پر ہے۔

(vi) پرنٹنگ کے لیے _____ شارٹ کٹ کمانڈ استعمال ہوتی ہے۔

(vii) MS ورڈ میں پہلے سے موجود فونٹ سائز _____ پوائنٹ ہے۔

(viii) CTRL+C _____ کے لیے استعمال ہوتا ہے۔

(ix) دیسارس، سائنومز کو دیکھنے اور _____ کے لیے استعمال ہوتے ہیں۔

(x) Office Assistant _____ پہلے سے موجود ہے۔

2۔ درست جواب کا انتخاب کریں۔

(i) ڈبل انڈر لائن کے لئے کون سا شارٹ کٹ استعمال ہوتا ہے؟

(a) CTRL+SHIFT+D (b) CTRL+[(c) CTRL+SHIFT+M (d) SHIFT+F3

(ii) MS ورڈ _____ بیس پروگرام ہے۔

(a) ونڈوز (b) سسٹم (c) ڈاس (d) ان میں سے کوئی نہیں

(iii) مندرجہ ذیل میں سے کون سی بار ایپلیکیشن سوفٹ ویئر مہیا کرتی ہے؟

(a) مینیو بار (b) ٹول بار (c) سٹیٹس بار (d) سکرول بار

(iv) CTRL+2 کیز کو پریس کریں _____ جگہ کے لیے۔

(a) سنگل (b) ڈبل (c) ٹرپل (d) اِن میں سے کوئی نہیں

(v) ورڈ میں موجودہ ایکٹو ڈاکیومنٹ پرنٹ کے لیے _____ پرنٹ آئی کن کو کلک کریں۔

(a) مینیوئلی (b) آٹومیٹک (c) (a) اور (b) دونوں (d) اِن میں سے کوئی نہیں

(vi) پیراگراف کو سلیکٹ کرنے کے لیے مندرجہ ذیل میں سے کیا استعمال ہوتا ہے؟

(a) سنگل کلک (b) ڈبل کلک (c) رائٹ کلک (d) اِن میں سے کوئی نہیں

(vii) CTRL+Y استعمال ہوتا ہے۔

(a) اَن ڈو کے لیے (b) ڈھونڈنے کے لیے (c) پیج بریک کے لیے (d) ری ڈو کے لیے

(viii) مندرجہ ذیل میں سے کونسا آپشن ایڈٹ مینیو میں نہیں ہوتا؟

(a) اَن ڈو (b) ری ڈو (c) فائنڈ (d) فونٹ

(ix) سلیکٹڈ ورڈ کو بولڈ کرنے کے لیے کون سا شارٹ کٹ استعمال ہوتا ہے؟

(a) CTRL+SHIFT+B (b) SHIFT+B (c) ALT+B (d) CTRL+B

3۔ صحیح اور غلط کی نشاندہی کریں۔

(i) آمدن، اخراجات کے شمار کے لیے، بیلنس شیٹ وغیرہ کے لیے ایک ورڈ پروسیسنگ پروگرام استعمال ہوتا ہے۔

(ii) ورڈ پروسیسر کو الیکٹرونک ٹائپ رائٹر بھی کہتے ہیں۔

(iii) CTRL+U سلیکشن کو انڈر لائن بھی کرتا ہے۔

(iv) سٹیٹس بار، ایپلیکیشن ونڈو کے اوپر ہوتا ہے۔

(v) پرنٹ سیٹ صرف پرنٹر سلیکٹ کرنے دیتا ہے۔

(vi) CTRL+X سلیکشن کو کٹ کرتا ہے۔

(vii) ایک ڈاکیومنٹ کا نارمل ویو ہیڈر اور فٹر کی انفارمیشن نہیں دکھاتا۔

(viii) ایک ڈراپ کیپ ایک بڑا حرف ہے جو کہ ایک پیراگراف کو شروع کرتا ہے اور ٹیکسٹ کی کئی لائنز تک ڈراپ ہوتا ہے۔

(ix) MS ورڈ میں F2 کی ہیلپ کے لیے استعمال ہوتی ہے۔

(x) ڈاکیومنٹ میں جلدی سے سپیلنگ چیک کرنے کے لیے سپیل چیک استعمال ہوتا ہے۔

4۔ ٹائٹل بار سے کیا مراد ہے؟

5۔ MS ورڈ میں کلپ بورڈ کا استعمال بیان کریں۔

6۔ آپ اپنے ٹول بارز کو کیسے کسٹمائز کریں گے؟

7۔ ڈراپ کیپ سے کیا مراد ہے؟

8۔ فارمیٹنگ ٹول بار اور سٹینڈرڈ ٹول بار میں فرق بیان کریں۔

9۔ ڈاکیومنٹ کے مختلف ویوز کو بیان کریں۔

10۔ MS ورڈ کا سپیل چیکر کیسے استعمال کیا جاتا ہے؟

11۔ پرنٹ ویو کا کیا فائدہ ہے؟

12۔ ایک ڈاکیومنٹ کے مارجنز کو تبدیل کرنے کے کتنے طریقے ہیں؟

13۔ MS ورڈ میں الائنمنٹ اور انڈینٹ میں فرق بتائیں۔

جوابات

-1
(i) Letter (ii) ٹائپ فیس (iii) CTRL+S
(iv) 9 (v) ٹاسک بار (vi) CTRL+P
(vii) 12 (viii) کاپی (ix) antonyms
(x) ڈسپلیڈ

-2
(i) a (ii) a (iii) c
(iv) b (v) c (vi) b
(vii) b (viii) d (ix) d

-3
(i) غلط (ii) صحیح (iii) صحیح
(iv) صحیح (v) غلط (vi) صحیح
(vii) صحیح (viii) صحیح (ix) غلط
(x) صحیح

جوابات

1-	(i) Letter	(ii) ٹائپ فیس	(iii) CTRL+S
	(iv) 9	(v) ٹاسک بار	(vi) CTRL+P
	(vii) 12	(viii) کاپی	(ix) antonyms
	(x) ڈسپلیڈ		
2-	(i) a	(ii) a	(iii) c
	(iv) b	(v) c	(vi) b
	(vii) b	(viii) d	(ix) d
3-	(i) غلط	(ii) صحیح	(iii) صحیح
	(iv) صحیح	(v) غلط	(vi) صحیح
	(vii) صحیح	(viii) صحیح	(ix) غلط
	(x) صحیح		

اصطلاحات

ٹاپ ڈاؤن ڈیزائن: مجموعی طور پر بڑے مسئلہ پر توجہ مرکوز کرنے کی بجائے ہم ہر تحتی مسئلہ کو الگ سے حل کرنے کی کوشش کرتے ہیں۔ اس سے سادہ حل نکل آتا ہے۔ یہ حکمتِ عملی ٹاپ ڈاؤن ڈیزائن (تقسیم کرو اور فتح کرو کا اصول بھی) کہلاتی ہے۔

ڈیسک چیکنگ: ڈیسک چیکنگ الگورتھم ڈیزائن کا ایک اہم حصہ ہے جس کو اکثر نظر انداز کر دیا جاتا ہے۔ ایک الگورتھم کو ڈیسک چیک کرنے کے لیے، ہمیں الگورتھم کے ہر ایک مرحلہ پر اس طرح عمل کرنا چاہیے جس طرح کہ ایک کمپیوٹر کرتا ہے، اور تصدیق کرنی چاہیے کہ الگورتھم حسب منشا کام کرتا ہے۔

پروگرام: ایک پروگرام کسی خاص مسئلہ کو حل کرنے کے لیے کمپیوٹر کو دی گئی ہدایات کا مجموعہ ہوتا ہے۔

ڈیسک چیکنگ: ڈیسک چیکنگ کسی ٹیسٹ ڈیٹا کی مدد سے کاغذ پر الگورتھم کے کام کے محتاط مشاہدہ کا عمل ہے۔ الگورتھم کو مختلف قیمتوں کی شکل میں ان پٹ دیا جاتا ہے جس کے آؤٹ پٹ کا جائزہ لیا جاتا ہے۔

سینٹیکس: کسی پروگرام لینگویج میں پروگرام لکھنے کے اصول اس پروگرامنگ لینگویج کا سینٹیکس کہلاتے ہیں۔

ڈی بگنگ: پروگرام میں ایررز تلاش کرنے اور دور کرنے کے عمل کو ڈی بگنگ کہتے ہیں۔

سینٹیکس ایررز: جب پروگرام، پروگرام لینگویج کے ایک یا زائد گرائمر کے اصولوں کی خلاف ورزی کرتا ہے تو سینٹیکس کی ایرر واقع ہو جاتی ہے۔

رن ٹائم ایررز: جب پروگرام کمپیوٹر کو کوئی غیر قانونی یا غیر تعریف شدہ کام کرنے کی ہدایت دیتا ہے تو رن ٹائم ایرر واقع ہو جاتی ہے، جیسا کہ کسی عدد کو صفر سے تقسیم کرنا۔

منطقی ایررز: جب پروگرام ایک غلط الگورتھم کی پیروی کرتا ہے تو منطقی ایرر واقع ہو جاتی ہے۔

پروگرام لاگو کرنا: ایک مرتبہ پروگرام مکمل طور پر ٹیسٹ ہو جانے کے بعد ایسی جگہ انسٹال (Install) کرنا یا رکھنا چاہیے جہاں اسے استعمال کیا جائے گا۔ اس مرحلہ کو پروگرام کا عملی استعمال کہتے ہیں۔

الگورتھم: الگورتھم مراحل کا ایک متناہی سیٹ ہے جس کی اگر پیروی کی جائے تو ایک خاص کام تکمیل تک پہنچتا ہے۔ الگورتھم واضح، حتمی اور مؤثر ہونا چاہیے۔

فلوچارٹ: فلوچارٹ الگورتھم کا بذریعہ تصاویر اظہار ہے۔

ڈائریکٹ موڈ: ڈائریکٹ موڈ میں، جی ڈبلیو-بیسک کی کمانڈ ٹائپ کرتے ہی ایگزیکیوٹ ہو جاتی ہیں۔

اِن ڈائریکٹ موڈ: اِن ڈائریکٹ موڈ پروگرام ٹائپ کرنے کے لیے استعمال ہوتا ہے۔ پروگرام سٹیٹمنٹس کے شروع میں لائن نمبر دیے جاتے ہیں اور یہ میموری (Memory) میں سٹور ہو جاتی ہیں۔

IDE: (Integrated Development Environment)

پروگرام لوڈ کرنا: پروگرام کو لوڈ کرنے کا مطلب اس کو سیکنڈری سٹوریج (Secondary storage) آلہ (جیسا کہ ہارڈ ڈسک Hard disk) سے میموری میں لانا ہے تاکہ اس کی ہدایات پر عمل کیا جا سکے۔

پروگرام چلانا: پروگرام چلانے سے مراد پروگرام میں ہدایات پر عمل کرنا ہے۔ پروگرام چلانے سے پہلے اسے میموری میں لوڈ کرنا چاہیے۔

ذخیرہ الفاظ: ذخیرہ الفاظ یا کی ورڈز (Keywords) ایسے الفاظ ہیں جن کا مطلب بیسک میں پہلے ہی بیان کر دیا گیا ہے۔ ان کا استعمال پہلے سے ہی طے شدہ ہے اور یہ پروگرام میں کسی اور مقصد کے لیے استعمال نہیں ہو سکتے اور نہ ہی ان کے استعمال کا مقصد تبدیل کیا جا سکتا ہے۔

متغیرات: متغیرات کو میموری لوکیشنز (Memory locations)/ میموری سیلز (Memory cells) کا نام دیا جاتا ہے جن کو پروگرام کے ان پٹ ڈیٹا کو محفوظ کرنے کے لیے اور پروگرام کی ایگزیکیوشن کے دوران کمپیوٹیشنل نتائج کو ذخیرہ کرنے کے لیے استعمال کیا جاتا ہے۔

نومیرک متغیرات: نومیرک متغیرات نومیرک قیمتوں کو ذخیرہ کر سکتے ہیں۔ (نومیرک قیمتوں میں فلوٹنگ پوائنٹ (Floating point) نمبرز اور مکمل اعداد دونوں شامل ہیں)۔

سٹرنگ متغیرات: ایک سٹرنگ سے مراد ڈبل کوٹیشن میں بند ترتیب وار کریکٹرز ہیں۔ ایک سٹرنگ متغیر ترتیب وار کریکٹرز کو سٹور کر سکتا ہے۔

کانسٹینٹ: کانسٹینٹ ایک ایسی مقدار ہے جس کی قیمت تبدیل نہیں ہو سکتی۔

نومیرک کانسٹینٹس: نومیرک کانسٹنٹ انٹیجر، ایک لفظی یا دو لفظی اعداد پر مشتمل ہوتے ہیں۔ انٹیجر کانسٹینٹ ایسی قیمتوں کو ظاہر کرتے ہیں جنہیں گنا جاتا ہے اور ان میں کسری حصہ نہیں ہوتا۔

سٹرنگ کانسٹینٹس: ایک سٹرنگ کانسٹینٹ ڈبل کوٹیشن مارکس میں بند ترتیب وار نومیرک کریکٹرز پر مشتمل ہوتا ہے۔ سٹرنگ کانسٹینٹ کی زیادہ سے زیادہ لمبائی 255 کریکٹرز ہوتی ہے۔

اَرتھ میٹک اوپریٹرز: اَرتھ میٹک اوپریٹرز، اعداد پر اَرتھ میٹک عوامل انجام دینے کے لیے اِستعمال کیے جاتے ہیں۔

ریلیشنل اوپریٹرز: ریلیشنل اوپریٹرز، دو قیمتوں کا موازنہ کرنے کے لیے اِستعمال کیے جاتے ہیں۔

منطقی اوپریٹرز: منطقی اوپریٹرز سادہ کنڈیشنز (Conditions) کی مدد سے پیچیدہ کنڈیشنز بنانے میں مدد دیتے ہیں (کنڈیشن سے مُراد، ایک ایکسپریشن ہے جو درست یا غلط کی نشاندہی کرے)۔

(Concatenation Operators): سٹرنگ ملانے کے آپریشن کے لیے جمع کی علامت '+' استعمال ہوتی ہے اور یہ دو سٹرنگز کو ملاتی ہے۔

اسائنمنٹ اوپریٹر: اسائنمنٹ اوپریٹر متغیر میں قیمت، سٹرنگ یا کمپیوٹیشنل نتیجہ کو ذخیرہ کرنے کے لیے استعمال ہوتا ہے۔

ٹائپ کنورژن: جب آپ کا پروگرام ایک قسم کی عددی قیمت کو دوسری تقسیم کے متغیر میں سٹور کرنے کی کوشش کرتا ہے، تو GW-BASIC مندرجہ ذیل اُصولوں کے مطابق ٹائپ کی تبدیلی (ٹائپ کنورژن) کا کام سرانجام دیتا ہے۔

کنٹرول سٹرکچرز: کنٹرول سٹرکچرز پروگرام پر عمل کو کنٹرول کرتی ہیں۔

غیر مشروط ٹرانسفر: غیر مشروط کنٹرول کے ٹرانسفر میں پروگرام کنٹرول کسی شرط کے بغیر ایک مخصوص لائن یا ایک سے زیادہ لائنز کو چھوڑ کر کسی خاص لائن کو منتقل ہو جاتا ہے۔

مشروط ٹرانسفر: مشروط کنٹرول کے ٹرانسفر میں پروگرام کنٹرول کسی خاص شرط کے تحت ایک مخصوص لائن یا ایک سے زیادہ لائنز کو چھوڑ کر کسی خاص لائن کو منتقل ہو جاتا ہے۔

سلیکشن سٹرکچر: سلیکشن سٹرکچر چناؤ کرتا ہے کہ کونسی متبادل پروگرام سٹیٹمنٹ کو ایگزیکیوٹ کرنا ہے۔

لُوپس: لوپ سٹیٹمنٹس کے ایک سیٹ کی ایک حد تک یا کسی خاص شرط کے پورا ہونے تک دُہرانے کی اجازت دے سکتا ہے۔

نیسٹڈ لُوپ: ایک لُوپ (FOR یا WHILE) کے اندر ایک یا ایک سے زیادہ (FOR یا WHILE) لُوپس ہو سکتے ہیں۔ ایسے لُوپ کو نیسٹڈ لُوپ کہتے ہیں۔

ارے: ایک ارے متغیرات کا سیٹ ہے جو کہ ایک ہی قسم کے ڈیٹا کو سٹور کر سکتا ہے۔ ہر میموری لوکیشن میں ایک قیمت درج ہوتی ہے جو ارے کا رُکن کہلاتی ہے۔ ارے کے ارکان تک رسائی اُن کی پوزیشن یا لوکیشن نمبر کے ذریعہ ہوتی ہے۔ اس پوزیشن نمبر کو انڈیکس (Index) یا سب سکرپٹ (Subscript) کہتے ہیں۔

لینئر ارے: یک سمتی ارے کو لینئر (Linear) ارے یا ویکٹر (Vector) ارے بھی کہتے ہیں۔ یہ صرف ایک قطار (Row) یا ایک کالم (Column) پر مشتمل ہوتا ہے۔

دو سمتی ارے: دو سمتی ارے قطاروں اور کالموں پر مشتمل ہوتا ہے۔ اسے جدول (table) یا قالب (matrix) بھی کہتے ہیں۔

سب پروگرامز: ایک الگ بڑے پروگرام کو چھوٹے اور مینیج ایبل (Manageable) حصوں میں بانٹ دیا جاتا ہے، جنہیں سب پروگرام یا ماڈیولز

(Modules) کہا جاتا ہے۔

سٹینڈرڈ فنکشنز: سٹینڈرڈ فنکشنز بیسک لینگویج کے ساتھ مہیا کیے جاتے ہیں اور استعمال کے لیے ان کا نام استعمال کرنا پڑتا ہے۔

بلٹ اِن فنکشن: یہ فنکشنز دی گئی قیمتوں (operands) پر عوامل (operations) سرانجام دیتے ہیں اور نتائج مہیا کرتے ہیں۔

نومیرک فنکشنز: ایسے فنکشنز کا صرف نومیرک ویلیوز پر ہی اطلاق کیا جاسکتا ہے اور ان کی مدد سے نومیرک نتائج حاصل کیے جاسکتے ہیں۔

سٹرنگ فنکشنز: سٹرنگ فنکشنز کریکٹر، سٹرنگز کو پروسیس کرنے کے لیے استعمال ہوتے ہیں اور ان کی مدد سے نومیرک ویلیوز یا سٹرنگ ویلیوز کی شکل میں جواب ملتا ہے۔

یوزر۔ ڈیفائنڈ فنکشنز: جو فنکشنز ہم لکھتے ہیں اُنہیں یوزر ڈیفائنڈ فنکشنز کہا جاتا ہے۔

سب روٹینز: ایک سب روٹین خودمختار سٹیٹمنٹس کے ایسے سیٹ پر مشتمل ہوتی ہے جسے کسی پروگرام میں کہیں سے بھی استعمال کیا جاسکتا ہے۔

کریکٹرز: کریکٹرز حروفِ تہجی، ہندسوں اور سپیشل کریکٹرز پر مشتمل ہوتے ہیں۔ انہیں کمپیوٹر میں ایک (1s) اور صفر (0s) کی مدد سے خاص ترتیب سے لکھا جاتا ہے۔

ڈیٹا فیلڈز: ڈیٹا فیلڈز متعلقہ حروف کے گروپ ہیں جو کہ خاص معلومات پر مشتمل ہوتے ہیں۔

ریکارڈ: متعلقہ فیلڈز کے گروپ کو ریکارڈ کہتے ہیں۔

سیقوئینشل ایکسیس: سیقوئینشل ایکسیس کا مطلب ہے کہ جو ڈیٹا مطلوبہ فائل میں محفوظ ہے اسے اس طرح سے ایکسیس کیا جائے گا جیسا کہ وہ ڈسک پر سٹور ہوا تھا۔ دوسرے لفظوں میں، اگر آپ ایک فائل کے پچیسویں (25th) ریکارڈ تک رسائی چاہتے ہیں تو پہلے ریکارڈ نمبر ایک تا چوبیس تک رسائی ہوگی اور پھر پچیسویں ریکارڈ تک رسائی ہوگی۔

رینڈم ایکسیس: رینڈم ایکسیس کے طریقہ میں ایک پروگرام کو ایک مخصوص ریکارڈ تک ڈائریکٹ رسائی (Direct access) حاصل ہو جاتی ہے۔ ظاہر ہے اس طرح سے ایک فائل سے ریکارڈ تلاش کرنا سیقوئینشل ایکسیس کے مقابلہ میں بہت آسان اور تیز تر ہو جاتا ہے۔

رینڈم فائلز: رینڈم فائلز ایک فائل میں تمام پچھلے ریکارڈز کو ایکسیس کیے بغیر مطلوبہ ریکارڈ کو ڈائریکٹ ایکسیس کر سکتی ہیں۔

گرافکس: معلومات کو تصویری شکل میں ڈیزائن کرنے اور پیش کرنے کے فن کو گرافکس کہتے ہیں۔

ریزولوشن: افقی (Horizontal) اور عمودی (Vertical) پکسلز کی تعداد سے مونیٹر کی ریزولوشن (Resolution) کا علم ہوتا ہے۔ ایک کوآرڈی نیٹ سے مراد ایک خاص پکسل (تصویری ایلیمنٹ: سکرین پر ایک نقطہ کو پکسل کہتے ہیں) ہے۔

ٹیکسٹ موڈ: یہ موڈ ٹیکسٹ ٹائپ ڈیٹا کے لیے استعمال ہوتا ہے۔ ٹیکسٹ بیسڈ گرافک میں سکرین پر ٹیکسٹ اور لائنیں کھینچی جاسکتی ہیں۔

میڈیم۔ ریزولوشن گرافک موڈ: میڈیم۔ ریزولوشن گرافک موڈ گرافک بنانے کے لیے استعمال ہوتا ہے۔ ڈسپلے سکرین کو 320×200 پکسلز کے میٹرکس میں تقسیم کیا جاتا ہے۔ اس طرح ہر ایک پکسل کی پوزیشن کا تعین سکرین کے ایکس اور وائی کوآرڈینیٹس سے کیا جاتا ہے۔

ہائی۔ ریزولوشن گرافک موڈ: ہائی۔ ریزولوشن گرافک موڈ 200 ×640 پکسلز کے میٹرکس میں گرافکس بنانے کے لیے استعمال ہوتا ہے۔

ورڈ پروسیسنگ: مائیکروسوفٹ ورڈ (MS Word)، ڈاکیومنٹس بنانے کے لیے ایک بے حد ضروری ٹول ہے۔

IRM: (Information Rights Management)

فونٹس: مائیکروسوفٹ ورڈ میں ٹیکسٹ شکل (typeface) بتانے کے لیے فونٹ کی اصطلاح استعمال ہوتی ہے۔

ڈراپ کیپس: ایک ڈراپ کیپ ایک بڑا حرف ہے جس سے پیراگراف شروع ہوتا ہے اور اس سے کافی زیادہ لائنز ڈراپ ہو جاتی ہیں۔

انڈیکس

نوٹ:

مطلوبہ لفظ کو اس کے پہلے حرف کے گروپ میں تلاش کریں۔