Programming

Chapter 5

101 COMP - 3

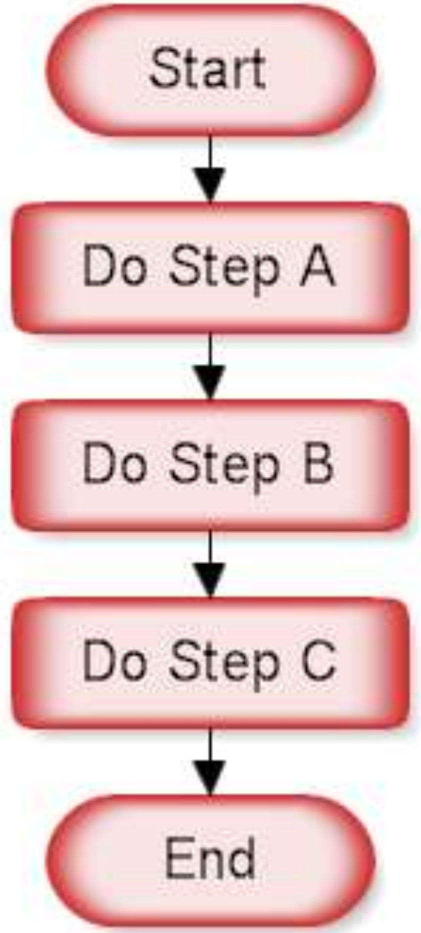
Algorithms

- Algorithm is a step by step method for solving a problem or doing a task.

Flowchart

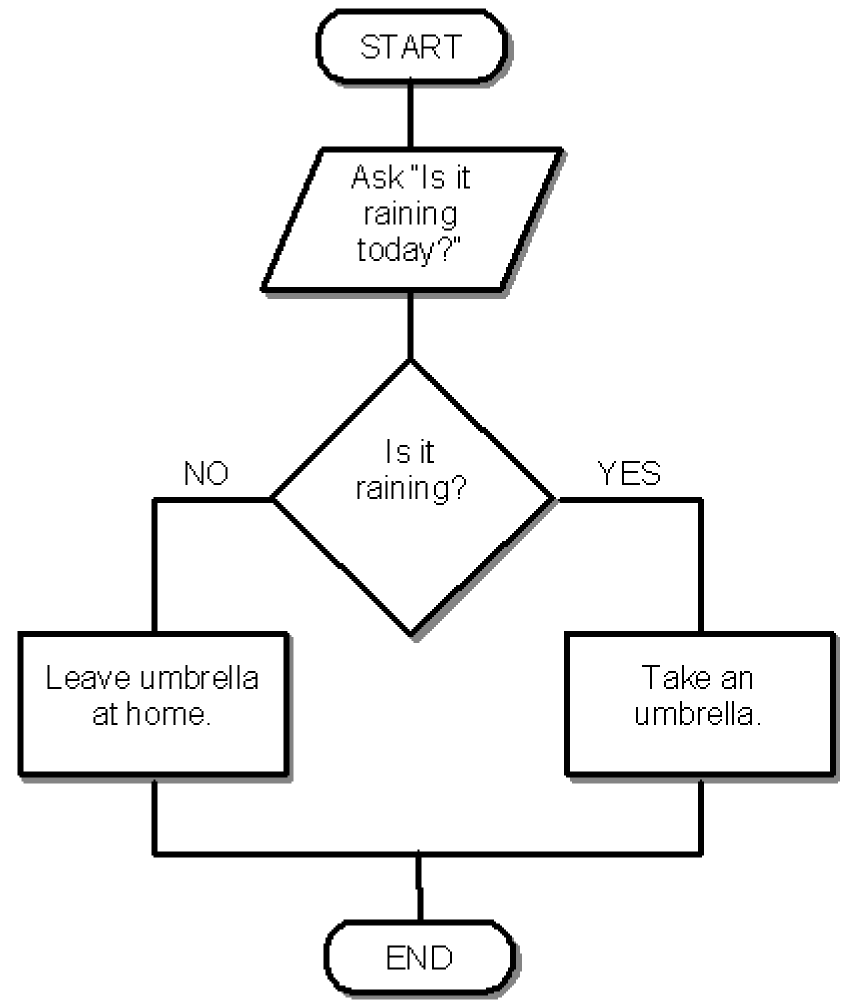
- Flowchart is a pictorial representation of an algorithm.

- Sequence.



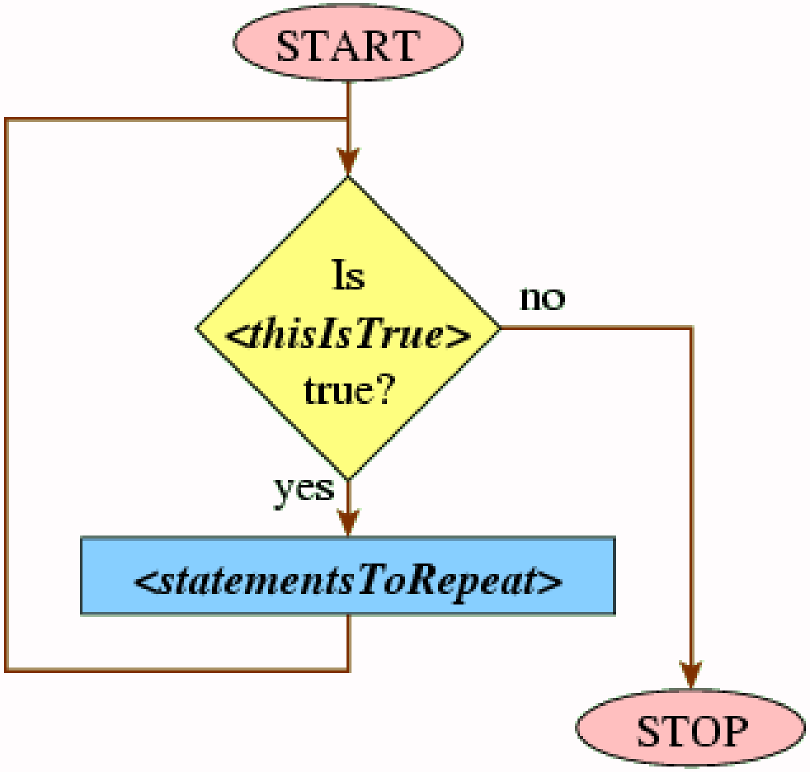
Flowchart

- Decision.



Flowchart

- Repetition.



Pseudocode

- Pseudocode: Is an english like representation of an algorithm.

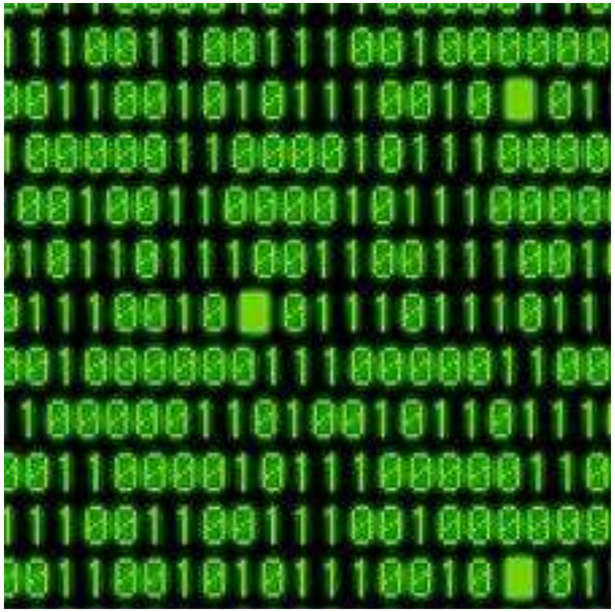
Programming Languages

- Computer language is a set of predefined words that are combined into a program according to predefined rules (syntax).

Machine Languages

- In the earliest days of computers, the only programming languages available were machine languages. Each computer has its own machine language, which is made of streams of 0s and 1s.

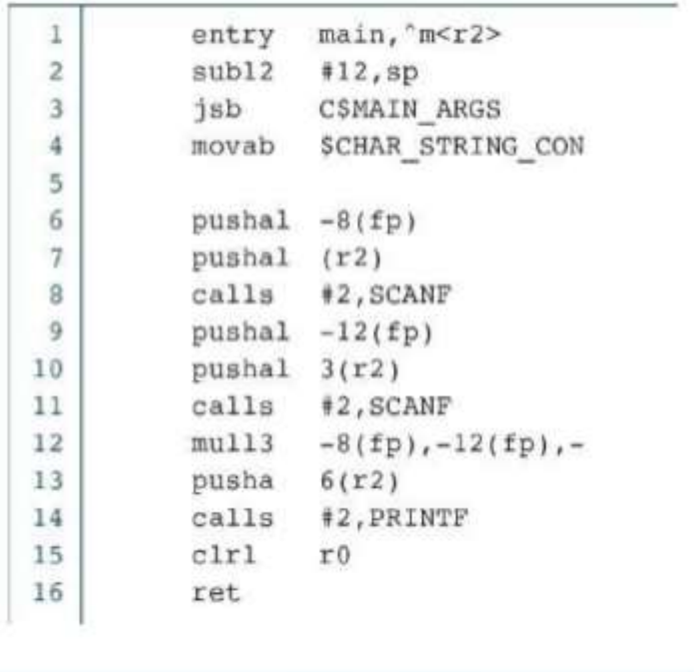
- The instructions in machine language must be in streams of 0s and 1s because the internal circuit of a computer is made of switches, transistors, and other Electronic devices that can be in one of two states: off or on.



- The only language understood by a computer is machine language.

Symbolic Languages

- Grace Hopper, a mathematician and a member of the U.S. Navy, developed the concept of a language that simply mirrored the machine languages using symbols, or mnemonics, to represent the various machine language instructions. Because they used symbols, these languages were known as symbolic languages.



```
        entry   main,^m<r2>
        subl2   #12,sp
        jsb     C$MAIN_ARGS
        movab   $CHAR_STRING_CON

        pushal  -8(fp)
        pushal  (r2)
        calls   #2,SCANF
        pushal  -12(fp)
        pushal  3(r2)
        calls   #2,SCANF
        mull3   -8(fp),-12(fp),-
        pusha   6(r2)
        calls   #2,PRINTF
        clrl    r0
        ret
```

Symbolic Languages

- A special program called an assembler is used to translate symbolic code into machine language. Because symbolic languages had to be assembled into machine language, they soon became known as assembly languages. This name is still used today for symbolic languages that closely represent machine language.

High-level Languages

- Although symbolic languages greatly improved programming efficiency, they still required programmers to concentrate on the hardware they were using. Working with symbolic languages was also very tedious because each machine instruction had to be individually coded. The desire to improve programmer efficiency and to change the focus from the computer to the problem being solved led to the development of high-level languages.

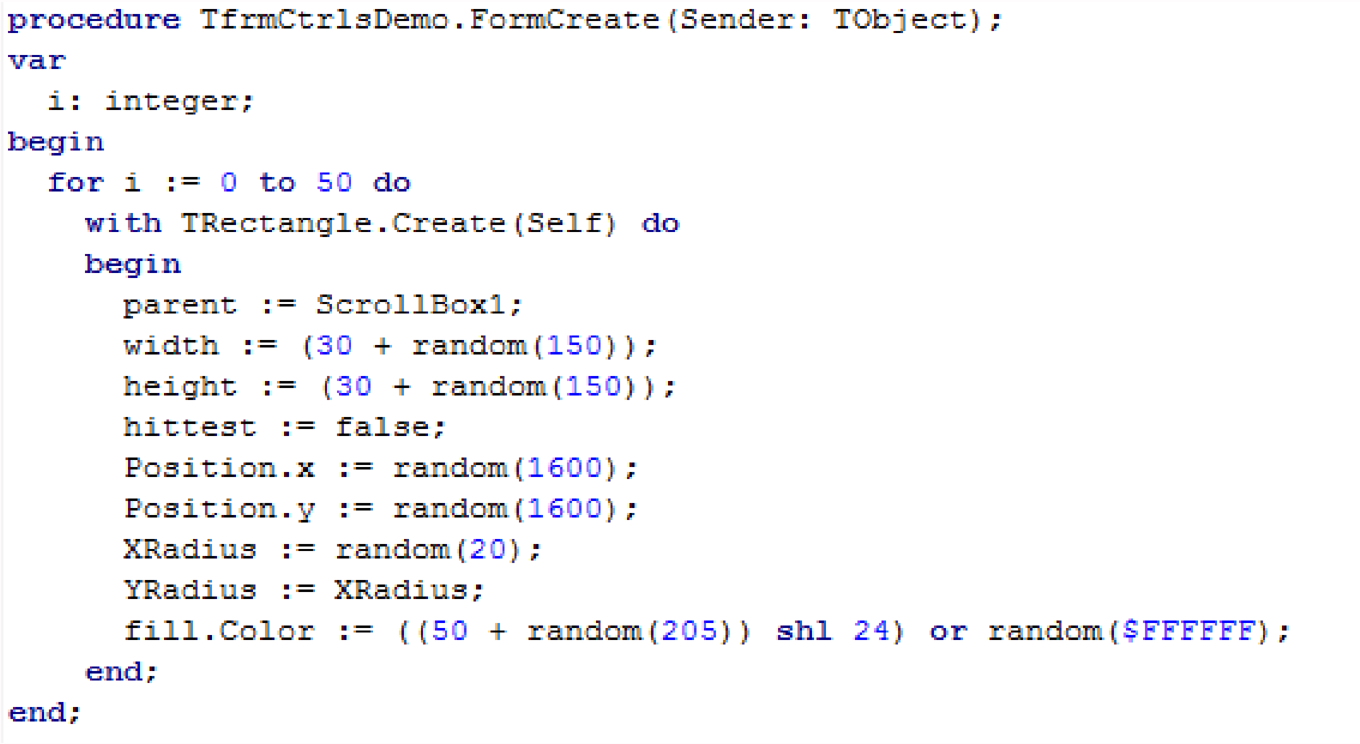
High-level Languages

- High-level languages are portable to many different computers, allowing the programmer to concentrate on the application rather than the intricacies of the computer. They are designed to relive the programmer from the details of assembly language.

- High-level languages share one characteristic with symbolic languages: They must be converted to machine language. This process is called compilation.

High-level Languages

- Over the years, various languages, most notably BASIC, COBOL, Pascal, Ada, C, C++, and Java, were developed.



```
procedure TfrmCtrlsDemo.FormCreate(Sender: TObject);
var
  i: integer;
begin
  for i := 0 to 50 do
    with TRectangle.Create(Self) do
    begin
      parent := ScrollBox1;
      width := (30 + random(150));
      height := (30 + random(150));
      hittest := false;
      Position.x := random(1600);
      Position.y := random(1600);
      XRadius := random(20);
      YRadius := XRadius;
      fill.Color := ((50 + random(205)) shl 24) or random($FFFFFF);
    end;
end;
```

Natural Languages

- Ideally, you could use your natural language (e. g., Engish, French, or Chinese), and the computer, would understand it and execute your requests immediately. Although this may sound like something out of science fiction, considerable work on natural languages is being done in labs today. So far, its use in industry is still quite limited.