Computer

Computer: is a an electronic device that is able to accept data as input, process that data using algorithms and data structures, and perform tasks as output. A computer consists of hardware and software.

1. **Hardware:** refers to the physical elements of a computer, for example monitor, keyboard, and mouse.

Hardware components:

1. **Central Processing Unit (CPU):** is brain of a computer. It is responsible for all functions and processes. The CPU is the most important element of a computer system.
2. **Memory:** parts of the computer that hold information.
   - ***The primary storage device*** *(Random Access Memory (RAM), and Read Only Memory (ROM)).*
   - ***The secondary storage device*** *(hard disk, CD, flash, ...).*
3. **Input devices:** parts of the computer that allow information or data to be given to the computer like: keyboard, mouse, scanner.
4. **Output devices:** parts of the computer that gives out information generated by the computer, like a monitor, printer, and speaker.

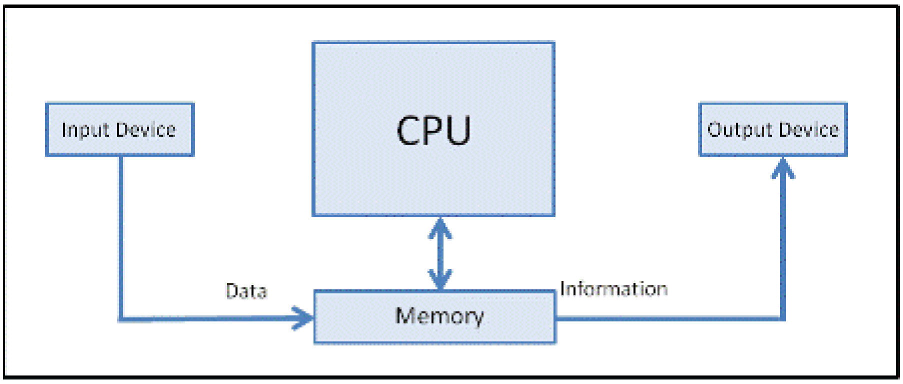


Figure (1): computer component.

2. **Software:** known as programs or applications consists of all the instructions that tell the hardware how to perform a task.

Software systems divided into three major groups:

1. **System software**: Helps run computer hardware and computer system. Computer software includes operating systems, device drivers, diagnostic tools and more.
2. **Programming software**: Software that assists a programmer in writing computer programs.
3. **Application software**: Allows users to accomplish one or more tasks.

Figure (2): hardware and software.

Comparison between hardware and software:

1. Hardware is a physical entity, while software is a non-physical entity.
2. Software is capable of performing many tasks as opposed to hardware which can only perform mechanical tasks that they are designed for.
3. Without software the hardware of a computer would have no function.
4. Without hardware (CPU) to perform tasks the software would be useless.

Programing language

Programing language: is a formal language which comprises a set of instructions used to produce various kinds of output. A computer performing some kind of computation or algorithm.

Programming languages differ from *natural languages*. The natural languages are only used for interaction between people, while programming languages involves a computer and allow humans to communicate instructions to machines.

Components of programming language

1. **Variables**: a variable is a storage location and an associated symbolic name which contains some known or unknown quantity or information, for example:

Dim *x* **as integer**

x = 10

x : this variable is a storage location must have integer value.

Dim *str* **as string**

Str = " welcome "

Str : this variable is a storage location defined to have string.

2. **Control Structures**: is a block of programming that analyzes variables and chooses a direction in which to go based on given parameters.

   1. **If Statement**

```
If (condition) then
        One or more Statement,
End if
```

2. **Select case**

```
Select case (value)
        Case 1: statement(s),
        Case 2: statement(s),
                .
                .
                .
        Case else: statement(s),
End select
```

3. **Data Structures:** is a particular way of storing and organizing data in a computer so that it can be used efficiently.

4. **Syntax**: the syntax of a programming language is the set of rules that define the combinations of symbols that are considered to be correctly structured programs in that language.

5. **Looping**: repeat set of instructions until some condition is met.

a.
```
For (number of times) do
        Statement(s)
Next
```

b.
```
While (condition)
        Statement(s)
 End while
```

c.
```
Do
Statement(s)
Until (condition)
```

Types of data structures

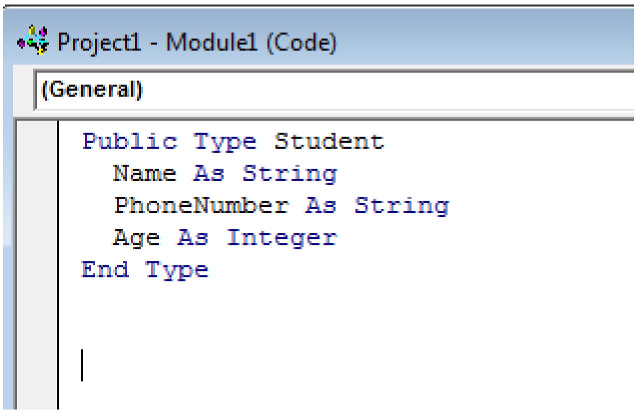
- **Primitive:** integer, real, boolean, string, etc.
- **Abstract:** complex data structures are used to store large and connected data, generally built upon simpler primitive data types for example:
    - ✓ Array,
    - ✓ Structure,
    - ✓ Linked List
    - ✓ Tree, etc.

Array

- Number of elements in a specific order.
- Elements are accessed using an integer index to specify which element is required.
- Homogeneous data structure: All the elements are of same type.
- Arrays may be fixed length or resizable.
- Example: dim A(3) **as integer** {40, 34, 28}

Structure

- Also called User-Defined data type because variables of different data types combined as a single variable to hold several related information.
- A structure must have at least one element.
- Non Homogeneous data structure: the elements may or may not be of the same type.
- Example:



```
Public Type Student
  Name As String
  PhoneNumber As String
  Age As Integer
End Type
```

```
Dim Stud As Student
While s <> " "
   s = InputBox("Enter Student name")
   Stud.Name = s
   Stud.PhoneNumber = "123456"
   Stud.Age = "21"
   List1.AddItem (Stud.Name)
   List1.AddItem (Stud.PhoneNumber)
   List1.AddItem (Stud.Age)
 Wend
```

Question 1: write a program to find the average of 10 numbers.

```
Dim x, sum, Avg As Integer
For i = 0 To 9
    x = InputBox("Enter integer number", "Input")
    sum = sum + x
Next
Avg = sum / 6
Text1.Text = Avg
```

Question 2: write a program to find the factor of number 5.

```
n = 5 : S = 1
While n > 0
     S = S * n
     n = n - 1
Wend
Text1.Text = S
```

Question 3: write a program to check the number if it is odd or even and print message.

```
Dim x As Integer
x = InputBox ("Input integer number ")
If x Mod 2 = 0 Then
    MsgBox "Even number"
Else
    MsgBox "Odd number"
End If
```

Question 4: write a program to define the range of a degree student as message.

```
x = InputBox(" Insert a degree ")
Select Case x
   Case  0 To 49: MsgBox "Fail"
   Case  50 To 59 : MsgBox "Average"
   Case  60 To 69 : MsgBox "Medium"
   Case  70 To 79 : MsgBox "Good"
   Case  80 To 89 : MsgBox "Very Good"
   Case  90 To 100 : MsgBox "Excellent"
End Select
```

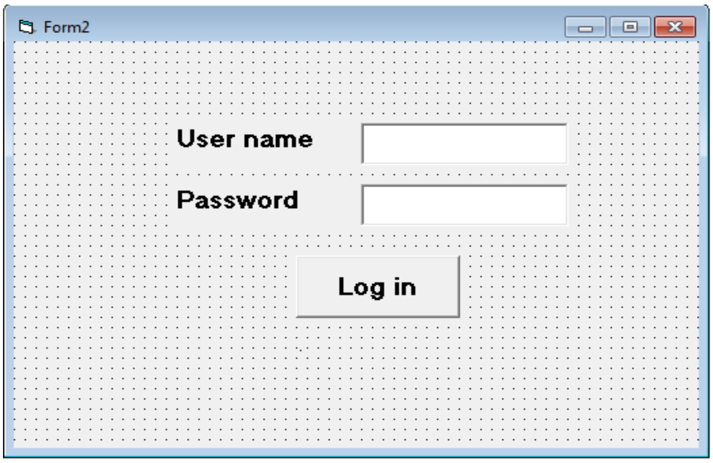
Another solution:

```
x = InputBox("Insert degree")
If x > 0 and x <= 49 Then
        MsgBox "Fail"
        ElseIf x >= 50 And x < 60 Then MsgBox "Average"
        ElseIf x >= 60 And x < 70 Then MsgBox "Medium"
        ElseIf x >= 70 And x < 80 Then MsgBox "Good"
        ElseIf x >= 80 And x < 90 Then MsgBox "Very Good"
        ElseIf x >= 90 And x <= 100 Then MsgBox "Excellent"
End if
```

Another solution:

```
x = InputBox("Insert degree")
Select Case x
   Case Is > 50: If x > 50 And x < 60 Then MsgBox "Average"
                 If x > 60 And x < 70 Then MsgBox "Medium"
                 If x > 70 And x < 80 Then MsgBox "Good"
                 If x > 80 And x < 90 Then MsgBox "Very Good"
                 If x > 90 And x <= 100 Then MsgBox "Excellent"
   Case Is < 50: MsgBox "Fail Fail "
End Select
```

Question 5: write a program to design a login window that show another form when user name and password true otherwise show an error message.



```
If (Text1.Text = "ABCDE") Then
  If (Text2.Text = "12345") Then
    Form2.Show
  Else
    MsgBox " Error Password, Try again "
  End If
Else
   MsgBox " User Name and/or password error, Try again "
End If
```

Question 6: write a program to find the maximum of 50 numbers.

```
Dim i, x, y As Integer
y = 0
For i = 1 To 50
    x = InputBox("Input number")
    If y < x Then
       y = x
    End If
Next
Form1.Print y
```

Question 7: write a program to find the minimum of 15 numbers.

```
Dim i, y, A(15) As Integer
For I = 0 to 14
   A(i) = InputBox("Input number")
```

```
Next

y = A(0)
For i = 1 To 14
   If y > A(i) Then
       y = x
   End If
Next
Form1.Print y
```

Question 8: write a program to full first list with student name, second list with student degree and print the name of passed student in third list.

```
For i = 0 To 9
  List1.AddItem (InputBox("Enter student name"))
  List2.AddItem (InputBox("Enter student degree"))
 Next
For i = 0 To 9
   If Val(List2.List(i)) > 50 Then
     List3.AddItem (List1.List(i))
   End If
Next
```

Question 9: write a program to search about specific number and when find it print its sequence.

```
Dim B(8) As Integer
B(0) = 200: B(1) = 250: B(2) = 300: B(3) = 350: B(4) = 400: B(5) = 450
B(6) = 500: B(7) = 550: B(8) = 600

X= Inputbox ("Enter number you search for")
For i = 0 To 8
   If B(i) = x Then
     Text1.Text = i
   End If
Next
```

Question 10: write a program to cut “world” word in the following string “hello world”

```
 Dim Str As String
 Str = InputBox ("Input string")
```

```
Text1.Text = Mid (Str, 7 , 5)
```

Question 11: write a program to concatenate two or more strings using + or **&** character.

```
Label1.Caption = Text1.Text + Text2.Text
Label2.Caption = Text1.Text & Text2.Text
```

Question 12: write a program to search about numbers less than 400 in first list, print them in second list, and print their sequence in third list.

```
For i = 0 To 8
   If B(i) < 400 Then
      List1.AddItem (B(i))
      List2.AddItem (i)
   End If
Next
```

Question 13: write a program to full array with 10 numbers and sort them in ascending order.

```
Dim A(10) As Integer
For i = 0 To 9
  A(i) = InputBox("Enter integer number")
Next

For i = 0 To 10
  For j = i + 1 To 9
     If Val(A(i)) > Val(A(j)) Then
        temp = A(i)
        A(i) = A(j)
        A(j) = temp
     End If
  Next
    Form1.Print A(i)
Next
```

Database

A database is a collection of information that is organized so that it can be easily to accessed relevant information, managed and updated.

Database table is composed of **records** and **fields** that hold data.

A single file with many records, each record have the same set of fields.

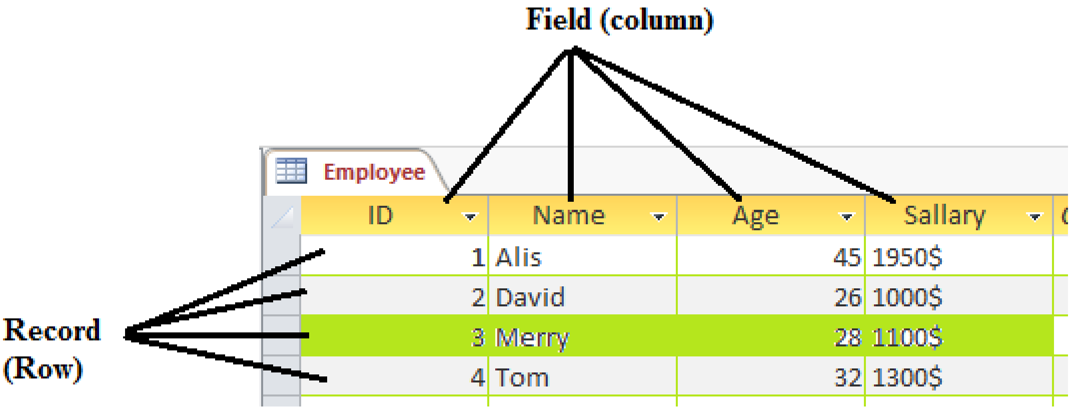
Record appear as **row** in the database table.

Field is part of a record and appear as **column** in the database table.

Example

In figure (1) employee database consist of fields (Id, Name, Age, Salary) and records {(1, Alis, 45, 1950$), (2, David, 26, 1000$), …}

Attribute component or data used in a GIS are stored in relational database tables.



ID	Name	Age	Sallary
1	Alis	45	1950$
2	David	26	1000$
3	Merry	28	1100$
4	Tom	32	1300$

Figure (1): employee database

Data Base Management System (DBMS) is a software makes it possible for users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and users or application programs, ensuring that data is consistently organized and remains easily accessible.

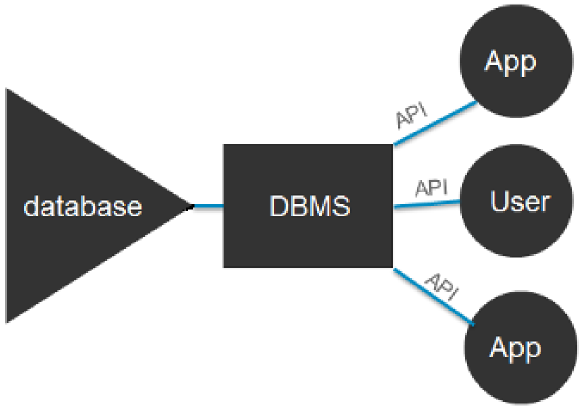


Figure (2): DBMS

Algorithm

An algorithm is a finite set of instructions, written in order, for solving a specific type of problems. Algorithm is **not** the complete code or program. It can be expressed either as high level description as **pseudocode** or using a **flowchart**.

Every Algorithm must satisfy the following **properties**:

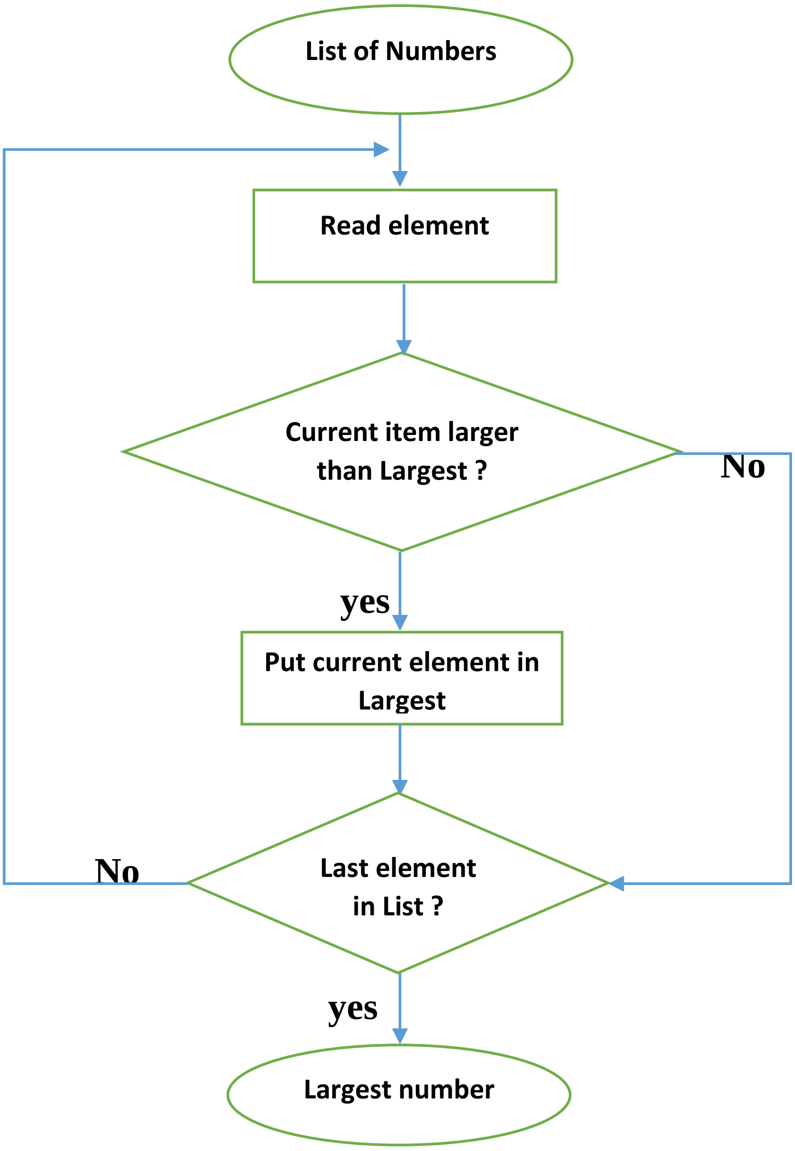
1. **Input:** There should be 0 or more inputs supplied externally to the algorithm.
2. **Output**: There should be at least 1 output obtained.
3. **Definiteness**: Every step of the algorithm should be clear and well defined.
4. **Finiteness**: The algorithm should have finite number of steps.
5. **Correctness**: Every step of the algorithm must generate a correct output.

Example: Write an algorithm and draw a flow chart to find the largest number in list.

Pseudocode

```
Algorithm Largest Number
  Input: A list of numbers L.
  Output: The largest number in the list L.
  largest ← L[0]
  for each item in L do
    if item > largest, then
      largest ← item
  return largest
```

Flow chart



Access Levels in Visual Basic

The access level of a declared element is the extent of the ability to access it, that is, what code has permission to read it or write to it. Code that cannot access a containing element cannot access any of its contained elements, even those declared as Public. For example, a Public variable in a Private structure can be accessed from inside subprogram that contains, but not from outside that subprogram.

The keywords that specify access level of the element's container are called **access modifiers.**

1. **Public:** specifies that the element can be accessed from code anywhere in the same project and from other projects.
2. **Private:** specifies that the element can be accessed only from within the same subprogram.
3. **Static:** Indicates that local variables of procedure are preserved between calls.

Note: The Dim statement without any access level keywords is equivalent to a Private declaration. However, you might want to use the Private keyword to make your code easier to read and interpret.

Functions and Procedures

Syntax

[**Public** | **Private** | **Friend**] [**Static**] *Procedure Name* (*argument1* **As** *type, argument2* **As** *type, ...*)

    [*statements*]

End procedure

Questions

1. Write a program to display the degree of student in the second list whose name typed in first text and exist in First list.

Solution:

```
List1.additem "Ali"
List1.additem "Ahmed"
List1.additem "Muna"
List1.additem "Nada"
List1.additem "Suha"

List2.additem "50"
List2.additem "90"
List2.additem "70"
List2.additem "65"
List2.additem "85"

For i=0 to 5
  If text1.text=list1.list(i) then
      Text2.text=list2.list(i)
   End if
Next
```

2. Write a program to sum two integer numbers every two positions.

Solution:

```
   List1.additem "50"
   List1.additem "90"
   List1.additem "70"
   List1.additem "65"
   List1.additem "85"
    i=0
    do while (i<6)
       x= list1.list(i)+list2.list(i+1)
       list2.additem (x)
       i=i+2
    loop
```

Another Way for solution

```
 For i=0 to 4
     If i mod 2 =0 then
        X= list1.list(i)+list2.list(i+1)
        list2.additem (x)
     end if
  next
```

3. Write a program to find the average of degrees larger than 80 and the students that have these degrees. (**Note**: **count** variable used to the count the number of students that have degrees >80 and s to sum these degrees).

Solution:

```
List1.additem "Ali"
List1.additem "Ahmed"
List1.additem "Muna"
List1.additem "Nada"
List1.additem "Suha"

List2.additem "50"
List2.additem "90"
List2.additem "70"
List2.additem "65"
List2.additem "85"

For i=0 to 4
  If list2.list(i) >80 then
    Form1.print List1.list(i)
    s=s+list2.list(i)
    count=count + 1
 end if
next
aveage= (s/count)
```

4. Write a program to isolate integers in array into even list for even numbers and odd list for odd numbers.

Solution:

```
Dim A(5) as integer
A(0)=328
A(1)=492
A(2)=281
A(3)=399
A(4)=286

For I = 0 to 4
    If I mod 2 =0  then
         List1.additem A(i)
    Else
        List2.additem A(i)
Next
```