Flow Charts and pseudocode

## Algorithms

Algorithms are the step-by-step process of performing a task. They are a set of instructions written to solve a problem. We use programming languages to implement the algorithms. They are very formal and Precise.

This makes it difficult for a non-programmer to understand. To make it easier, we use Flowcharts which are pictorial representations of the algorithms or Pseudocodes which consist of easy-to-understand statements on how the algorithm can be implemented.

## Flowcharts

Any computer program follows an algorithm that can be represented in terms of flowcharts or pseudocode. Flowchart is a pictorial representation of the code logic. This makes it easier for the programmer to write the code.

Below are the basic symbols of flow chart:

1. Start/End



1. Input/Output



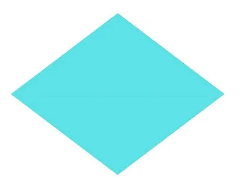
1. Process



1. Flow direction

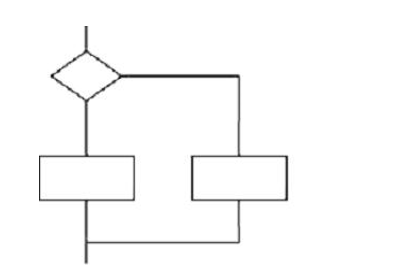


1. Decision

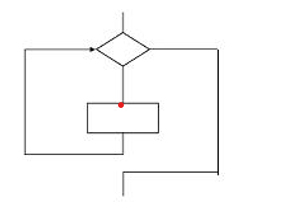


* Rectangles represent sequential statements and diamonds represent decisions.
* More than one sequential statement can be placed inside a single rectangle as long as there are no decisions in the sequence.
* Any decision is represented by a diamond, including those associated with loops.
* To create a flow chart representation of a complete program all we need to do is to connect together all the different bits of structure.

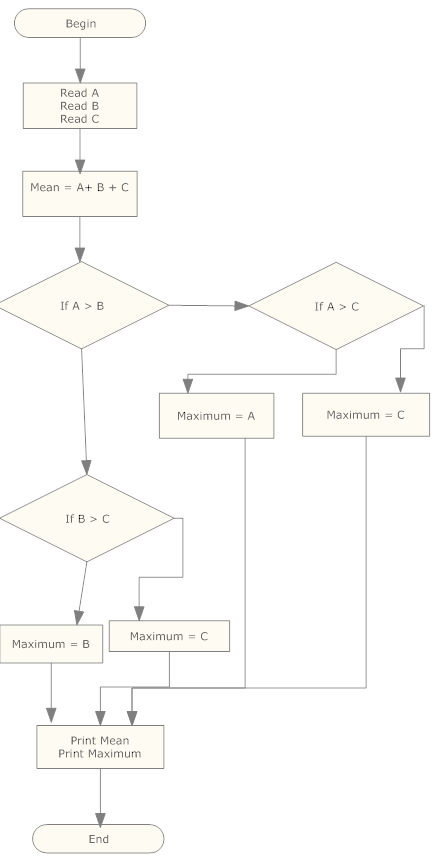
Example of a decision structure:



Example of a loop structure:



Simple example of a flow chart to print greatest of 3 numbers:

****

Example of flowchart with loops:

Find the sum of 5 number

**Start**

**Sum = 0 Count = 0**

Count

**End**

Yes

No

**Print Sum**

**Is Count <5 ?**

**Sum = Sum + n Count = Count + 1**

**Enter number n**

# Pseudocode

Pseudo code is a limited language than real programming language which enables designers to create all the main control structures needed by programs. It is sometimes used to document designs before they are coded into a programming language.

Real programming languages have a wide variety of forms and structures. The advantage of pseudo code in this respect is that it has a simple structure.

There are three main structures in pseudocode.

1. **Sequence:**

Here the statements are exercised one after the other as they appear on the page.   
  
Following is an example of a purely sequential program

1 Read A Enter a value for A : 1  
2 Read B Enter a value for B: 2  
3 C = A + B  
  
The computer would execute those three statements in sequence, so it would read (input) a value into A, then read another value into B, and finally add them together and put the answer into C.

1. **Selection:**

In this case the computer has to decide if a condition (known as a Boolean condition) is true or false.

If it is true the computer takes one route, and if it is false the computer takes a different route.

Selection structures therefore involve decisions.

Following is an example of a Selection program

1 IF Applicant\_Age >16  
2 THEN  
3 "Process License Application"  
4 ELSE  
5 "Come back when older"  
6 ENDIF

There may not always be an ELSE part, as below:

1 IF Applicant \_Age >16  
2 THEN  
3 "Process License Application"  
4 ENDIF

In this case the computer executes line 3 if the condition is true, or moves on to line 4 (the next line in sequence) if the condition is false

1. **Iteration:**

It simply involves the computer exercising a chunk of code more than once; the number of times it exercises the chunk of code depends on the value of a condition (just as in the selection case).

Iteration structures are called loops. The most common loop is known as a DO WHILE (or WHILE DO) loop and is illustrated below:

**Do While**

How many numbers between 15 and 20?

**1 X = 15  
2 Count = 0  
3 WHILE X < 20 DO  
4 X = X + 1  
5 Count = Count + 1  
6 END DO**

3 to 6 will be done until the condition is False.

X=15, Count=0 --> X = X+1--> X=16, Count=1

X=16, Count=1 --> X = X+1--> X=17, Count=2

X=17, Count=0 --> X = X+1--> X=18, Count=3

X=18, Count=0 --> X = X+1--> X=19, Count=4

X=19, Count=0 --> X = X+1--> X=20, Count=5

X=20 not less than 20... stop and come out of the loop.

As with the selection structures there is a decision.

In this case the condition that is tested at the decision is X < 20. --> True or False

**Repeat Until**

There is another variation of the loop structure known as a REPEAT UNTIL loop. It looks like this:

**1 X = 20  
2 Count = 0  
3 REPEAT  
4 X = X + 1  
5 Count = Count + 1  
6 UNTIL X < 20 True, False**The difference from a DO WHILE loop is that the condition is at the end, so the loop will always be executed at least once.

Every time the code inside the loop is executed the program checks the condition.

When the condition is true the program continues with the next sequential instruction. The outcome of this REPEAT UNTIL loop will be exactly the same as the DO WHILE loop a

**For Loop:**

This Loop is designed to execute for specific number of iterations. Often called as counting loop. Two Key words, FOR and ENDFOR are used. The General form is:

**FOR iteration bounds**

**Sequence**

**ENDFOR**

**Example:**

**FOR each month of the Year (good)**

**Print “month”**

**Select Case:**

**Case statement has the functionality of an IF-THEN-ELSE statement, except that multiple conditions can be specified.**

**A Select case statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being selected on is checked for each select case.**

**Select Case <test-expression>**

**Case condition\_1: result\_1**

**Case condition\_2: result\_2**

**…**

**Case condition\_n:result\_n**

**Case Else: result\_else**

**End Select**

**Example:**

**Input a Grade**

**Case based on Grade**

**Case=100:**

**Report “Perfect Score”**

**Case >89:**

**Report “Grade A”**

**Case >79:**

**Report “Grande B”**

**Case >69:**

**Report “Grade C”**

**Case >59:**

**Report “Grade D”**

**Default:**

**Report “Grade F”**

**End Case**

**Example:**

**Case Title Of**

**Mr : Print “Mister”**

**Mrs: Print “Missus”**

**Miss: Print “Miss”**

**Ms: Print “Mizz”**

**Dr: Print “Doctor”**

**Default: Mr**

**END CASE**

**Pseudocode to find greatest of 3 number:**

**Step 1: Input values of A, B and C**

**Steps 2: if (a>b) and (a>c) then**

**Largest = a**

**Else if (a>b) and (a<c) then**

**Largest = c**

**Else if (a<b) and (b>c) then**

**Largest = b**

**Else if (b>a) and (b<c) then**

**Largest = c**

**Step 3: Print largest**

Assignment:

Draw a flow chart and pseudo code for the Login system

Write pseudo code that performs the following: Ask a user to enter a number. If the number is between 0 and 10, write the word blue. If the number is between 10 and 20, write the word red. if the number is between 20 and 30, write the word green. If it is any other number, write that it is not a correct color option.

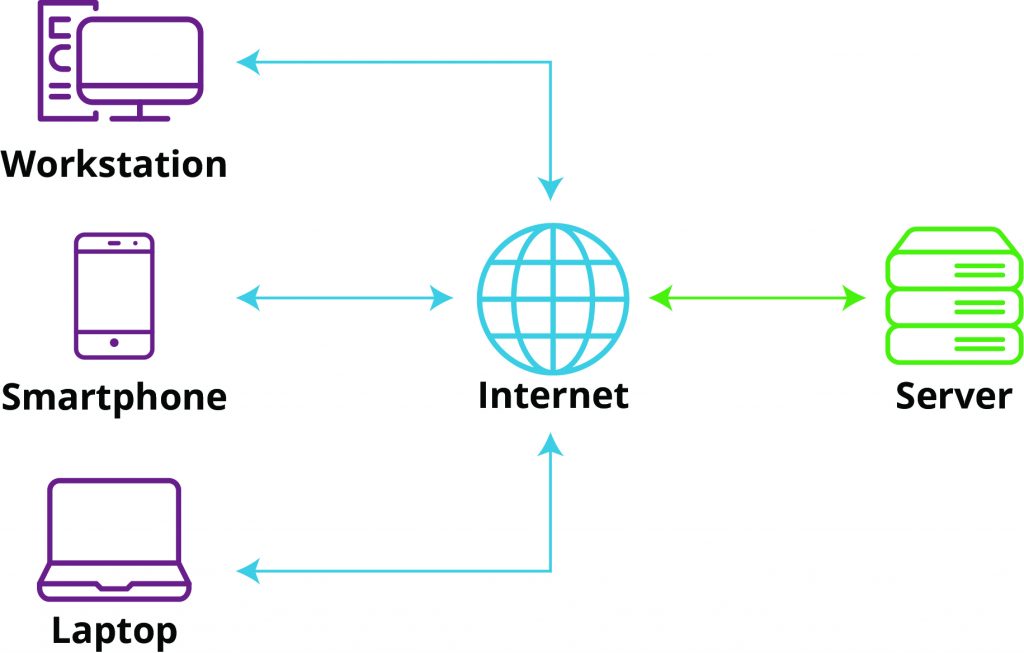
# Software Architectures

Software architecture is the organization of a system. This organization includes all components, how they interact with each other, the environment in which they operate, and the principles used to design the software.

## Client Server Architecture

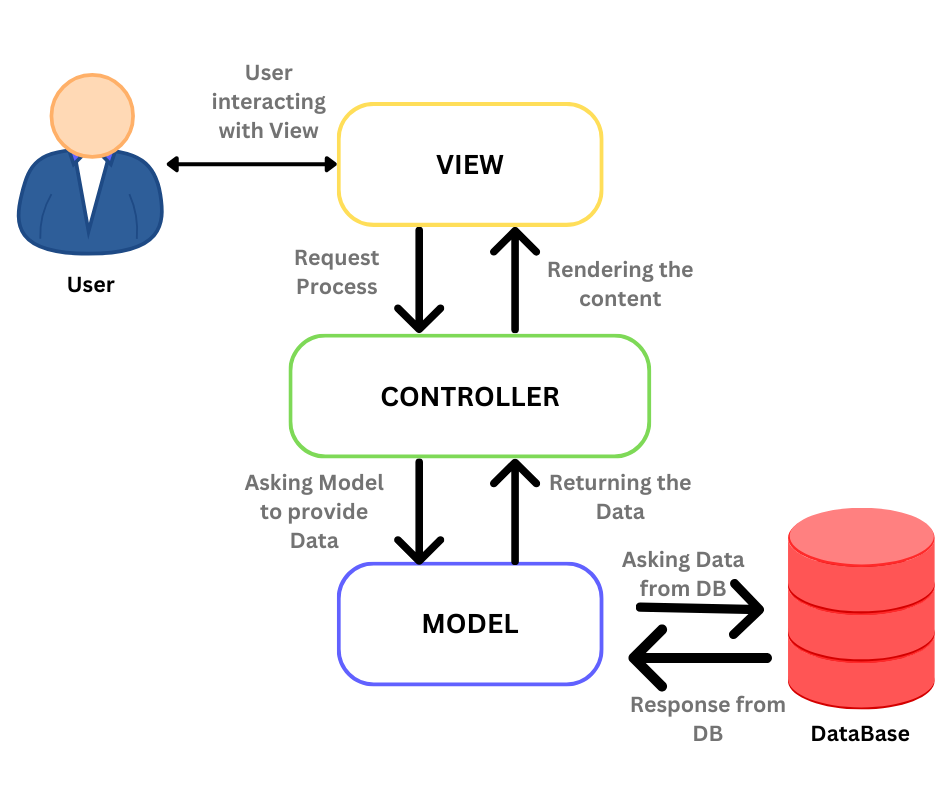
Client-server architecture facilitates communication between clients and servers regardless of whether they share the same network. A client requests data or services, and the server responds to the request by sending the resources.

Email is an example of the client-server design pattern. When you look for a particular email, the server searches its resource pool. Then, the server returns the requested email back to the client.



## Model-View Controller

The Model-View-Controller (MVC) architecture design is considered a [standard software development approach](https://www.techtarget.com/searchapparchitecture/tip/MVC-vs-MVVM-2-architecture-patterns-for-modularity) and is a backbone of popular development frameworks [like ASP.NET](https://www.techtarget.com/searchapparchitecture/tip/What-to-know-about-building-microservices-with-ASPNET-Core) and AngularJS. The model layer of MVC sits above the database and stores information about data types and underlying business logic. The view layer contains HMTL, CSS and other collections of front-end interface code. Finally, the controller provides a proxy between the model and the view elements to enforce the rules and methods surrounding data transactions. MVC is commonly used to build both web and mobile applications.



## Event Driven approach

Many programs stay on the lookout for certain events to occur to respond accordingly where the event could be triggered by certain data requests such as the selection of a button on a user interface (UI), the addition of an item to an online shopping cart, notification of payment on a point of sale (POS) system, etc. Event-driven architectures create, detect, consume, and react to events.

## Microservices

Any software is easy to manage if it's small, but large complex applications are often subjected to change and difficult to scale dynamically. Microservice architecture is the solution to this problem where software teams manage to create an architectural pattern consisting of small services for individual modules of application functionality.

Netflix was one of the first to adopt this approach to application development. The main UI Netflix users see is populated by collections of separate services, like the list of show history, account details, or personalized watch recommendations. So, Netflix's application is not just one program but a collection of multiple smaller programs.

This type of architecture is helpful when for instance, a collection of services requires more memory than others on average, then we can group these services into a domain where servers can always scale up or down quickly.

