2nd Semester

Unit - I

Page No.
Date

Features of C++

→ Object - Oriented Programming
It can create / destroy object while programming.
It can also create blueprint with which object can be created.

→ Machine Independent
A C++ executable is not platform independent they are machine independent.

→ Simple
It is a simple language, that programs can be broken down into logical units.

→ High - level language
C++ is a high level language.

→ Compiler Based
C++ is compiler based language that is used to compile the program and run.

OOP Vs Procedural Oriented Programming

→ OOP (Object - Oriented Programming)

- In OOP, the programs is divided into small parts called objects
- OOP follows a bottom - up approach
- Adding new data and function is easy
- OOP provide data hiding so it is more secure
- OOP is based on the real world
- In OOP, data is more important than function
- OOP is used for designing large and complex programs
- Ex - C++, Java

→ POP (Procedural Oriented Programming)

- In POP, the programs is divided into small parts called functions
- POP follows a top - down approach
- Adding new data and function is not easy
- POP does not provide proper way of hiding data so it is less secure
- POP is based on unreal world
- In POP, the function is more important than data
- POP is used for designing medium - size program
- Ex - C, Pascal etc.

OOP Concepts

→ Abstraction

Data abstraction is one of the most - important features of object - oriented programming in C++.

Abstraction means displaying only essential information and hiding the details.

Data abstraction refers to providing only essential information about data and hiding background details.

Consider a real life example of a man driving a car

- Types

→ Data abstraction

This type only shows the required information about the data and hides the unnecessary data

→ Control Abstraction

This type only shows the required information about the implementation and hides unnecessary information

- Advantages

1) Help user to avoid writing low - level language
2) Avoid code duplication and increases reusability.
3) Help to increase the security.

→ Inheritance

Inheritance is one of the most important feature of Object - Oriented Programing.

Inheritance is a feature or a process in which, new class are created from the existing classes.

The new class created is called "derived class" or "child class" and existing class is called "base class" or "Parent class".

- Mode of Inheritance

There are 3 mode of Inheritance

→ Public Mode
→ Protected Mode
→ Private Mode

- Types of Inheritance

→ Single inheritance
→ Multilevel inheritance
→ Multiple inheritance
→ Hierarchical inheritance
→ Hybrid inheritance

→ Polymorphism

The word "Polymorphism" means having many forms

We can define polymorphism as the abllity of a message to be displayed in more than one form.

Polymorphism is considered one of the important features of object - Oriented programming.

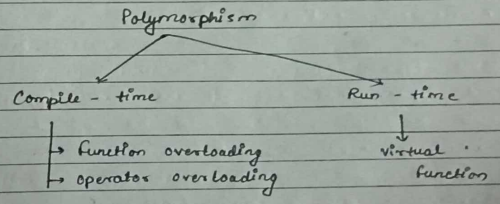
- Types

→ Compile - time Polymorphism

This type of polymorphism is achived by function overloading or operator overloading

→ Run - time Polymorphism

This type of polymorphism is achieved by function overriding.



→ Encapsulation

Encapsulation in C++ is defined as the wrapping up of data and information in a single unit.

- Encapsulation can be implemented using classes and access modifiers.
- In helps to control the modification our data members.
- Increase in the security of data.

→ Classes

The building block of C++ that leads to OOP is a class. It is a user-defined data type, which holds its own data members and member function.

A class is like a blueprint for an object.

Ex - Consider the class of cars.
There may be many cars with different names.

→ Objects

Basics of C++

→ Data type

- All variable use data type during declaration to restrict the type of data to be stored.
- Data types are used to tell the variable the type of data they can store.
- Every data type requires a different amount of memory.
- Data type divided into 3 types

→ Primitive data type or pre-defined
These data type can be used directly by the user to declare variables. Primitive data type available in C++.
- Integer • character • Boolean etc.

→ Derived data type
Derived data types that are derived from primitive or built-in datatypes. These can be of four types.
- function • Array • Pointer • Reference

→ Abstract or User defined data types
This types are defined by the user itself following user defined data type :-
- class • structure • Union • Enumeration

→ Variable

- Variable in C++ is a name given to a memory location. It is the basic unit of storage in a program.
- The value stored in the ~~program~~ variable can be changed during program execution.
- All the operation done on the variable effect that memo location.
- Types
  There are three types of variable

→ Local Variable
A variable defined within a block or method or constructor is called a local variable

→ Instance Variable
It is non-static variable and are declared in a class outside any method.

→ static Variable
Static variable are also known as class variable. It is declared using static keyword.

→ Uses of << >> Operators

- Left shift (<<)

It is a binary operator that takes two number, left shifts the bits of the first operand and second operand decided the number of places to shift.

Ex :- Left shifting an integer "a" with an integer "b" denoted as (a << b), multiplying a with 2^b.

- Right shift (>>)

It is a binary operator that takes two number, right shift the bits of the first operand and second operand decides the number of places to be shift.

Ex :- Right shifting an integer "a" with an integer "b" denotted as (a >> b), dividing a with 2^b.

Operators

An operator is a symbol that operates on a value to perform specific mathematical or logical computation.

An operator operands the operands

There are 6 types of operators in C++.

- Arithmetic operators
- Rational operators
- Logical operators
- Bitwise operator
- Assignment operator
- Conditional operator

Expression

- It consist of operators, constants and variable which are arranged according to the rule of language
- An expression can consist of one or more operands, zero or more operators
- An expression can be of following types

- Constant expression
- Integral expression
- Float expression
- Pointer expression
- Relational expression
- Logical expression
- Bitwise expression
- Assignment expression

Order of Evaluation

⑧ Decision Control

→ if

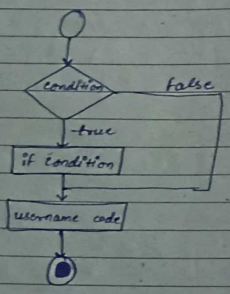
The if statement is the most simple decision making statement.

It is used to decide whether a certain statement or block of statement will be executed or not.

- Syntax

```
if (condition)
{
    // statement to execute if
    // condition is true
}
```

- Flowchart



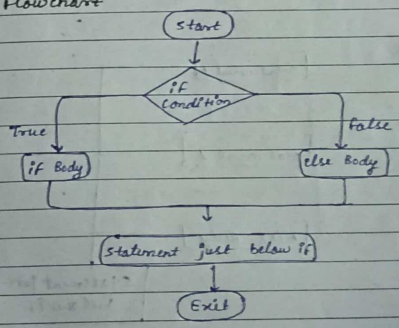
→ if - else

The if - else statement consists of two blocks, one for false expression and one for true expression.

- Syntax

```
if (condition)
{
    // Execute this block if
    // condition is true
}
else
{
    // execute this block if
    // condition is false
}
```

- Flowchart



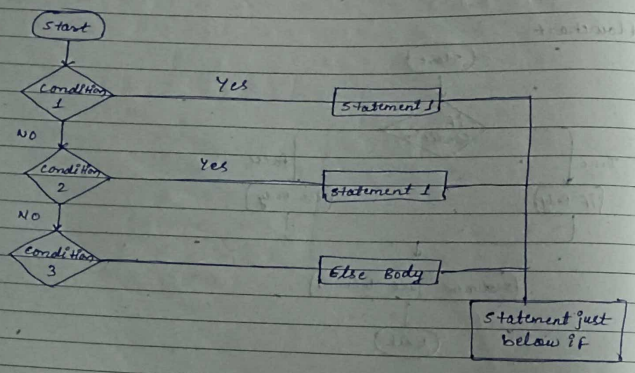
→ If - else - if

The if else if statement are used when the user has to decide among multiple options

- Syntax

```
if (condition)
    statement ;
else if (condition)
    statement ;
.
.
else
    statement ;
```

- flowchart



Loop Control

→ While

while loop in c++ is used in situation where we do not know the exact number of iteration of the loop beforehand.

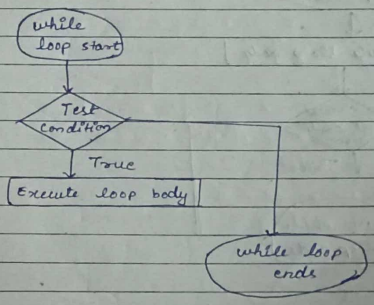
first cheack the condition then execute the body

- Syntax

```
while (test_expressions)
{
    // statement

    update_expression ;
}
```

- flow diagram



→ Do - while

Do - while loops come into use when we need to repeatedly execute a block of statement.

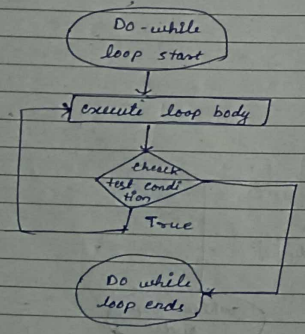
firstly, execute the body then condition check

- Syntax

```
do
{
  // loop body

  update - expression;
}
while (test - expression);
```

- Flow - chart



→ For

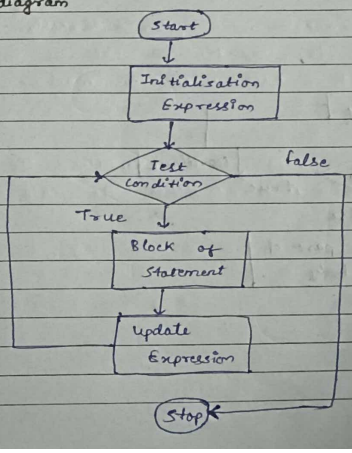
A for loop is a repetition control structure that allows us to write a loop that is executed a specific number of time

firstly initializes, then, condition check, execute body, update.

- Syntax

```
for
{
  // body of the loop
  // statements we want to execute
}
```

- Flow diagram



→ Break

The break in c++ is a loop control statement that is used to terminate the loop.

Break statements are used in situation when we are not sure about the actual number of iterations.

- Syntax

```
break;
```

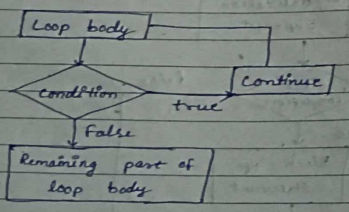
→ Continue

This loop statement is just like the break statement. The continue statement is opposite to that of the break statement.

- Syntax

```
continue;
```

- Flowchart



→ goto

The goto statement is used to jump from one point to another within a function.

- Flowchart