

OOPS with C++

Page No.		
Date		

Unit I

- 1) Introduction to programming concepts
- 2) Data types, Data (input, output), and operators
- 3) Decision making, loops, arrays & strings
- 4) Unified modelling language.
- 5) classes, abstraction, encapsulation

Unit II

- 1) Constructors and destructors
- 2) Working with objects
- 3) polymorphism
- 4) modeling relationships in class diagram

Unit III

- 1) Inheritance
- 2) modeling relationships
- 3) Runtime polymorphism
- 4) Pointers
- 5) File handling
- 6) Applying OOP to solve real life applications

Books:

- 1) Object Oriented programming with C++
author - Balagurusamy E 8th edition, McGraw Hill

1. Introduction to programming concepts

• Programming techniques

1) machine level programming language

2) Assembly level language programming

3) high level programming

1) Machine level programming language

Machine code or machine language is a set of instructions executed directly by a computer's CPU

2) Assembly language

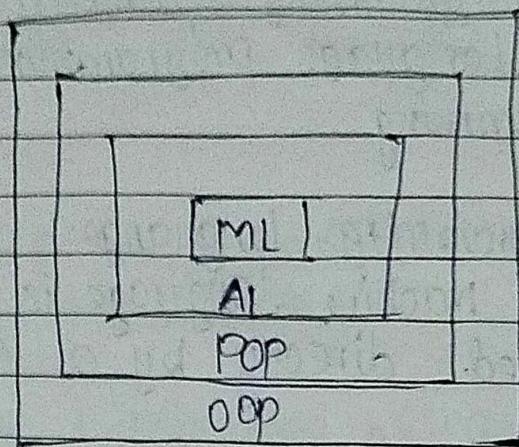
- It is a low level programming language for a computer or other programming device.
- There is very strong generally one-one correspondence between the language and the architecture's machine code instruction.
- Assembly language is converted into executable machine code or instruction by a utility program referred to as assembler.
- The conversion is known as assembling

3) High level language.

- High level language is any programming language that enables development of a program in much simpler programming context.
- It focuses more on programming logic rather than underlying hardware components memory addressing and register utilization.

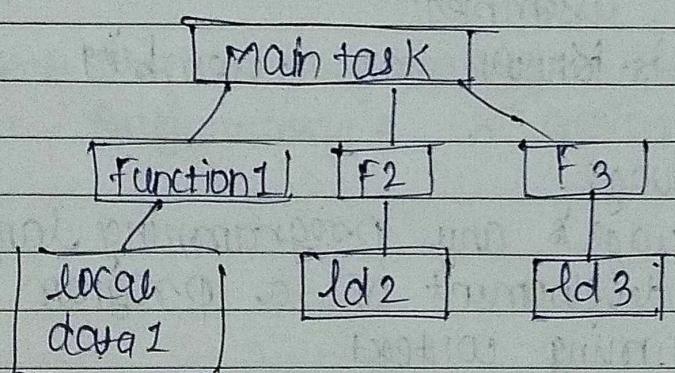
High level language is broadly classified into 2 types

- 1) Procedure Oriented Programming (POP)
- 2) Object Oriented Programming (OOP)



3/12/22

- 1) Procedure Oriented programming
 - It is conventional programming which uses high level languages such as COBOL, FORTAN, C
 - In POP the problem is viewed as a sequence of things to be done i.e. reading, calculating & printing
 - A no. of functions are written to accomplish this task, so the primary focus is on functions



Disadvantages:

- It does not model real world problems very well
- there is no data hiding

Analysis of POP

- concentration is on the development of functions, very little attention is given to the data that is being used by function
- Data moves freely between functions where there is chance for the intruder/hacker to hack the data.
- Critical application can't be design using this approach

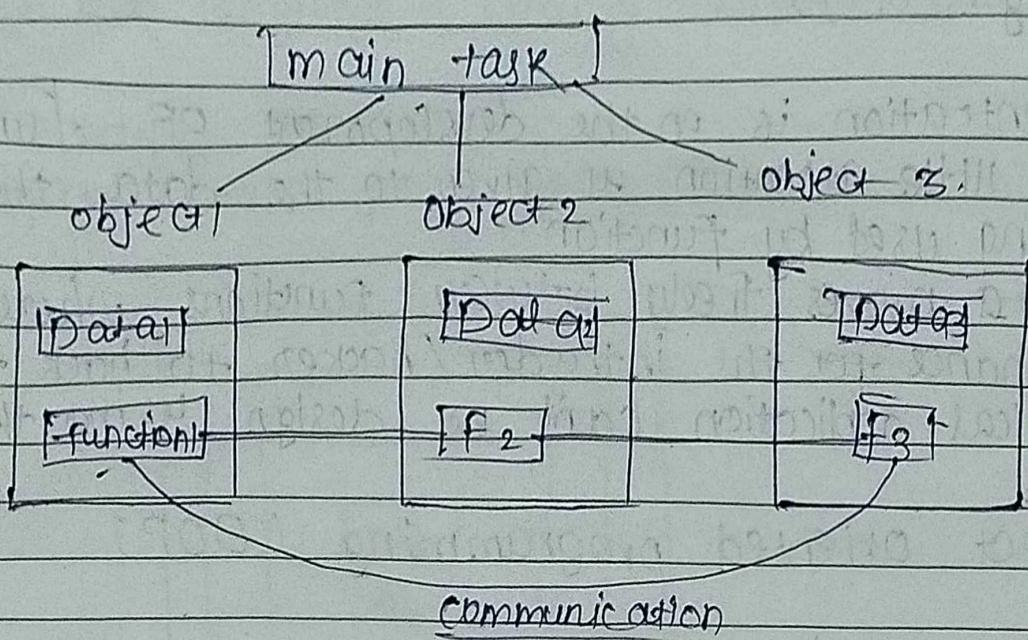
2) Object oriented programming (OOP)

OOP paradigm

- ↳ The measure motivating factor in the invention of OOP approach is to remove sum of flaws encountered in the procedure approach
- ↳ OOP treats data as a critical element in the program development and doesn't allow data to flow freely around the system
- ↳ It ties the data more closely to the functions that operates on it and protects the data from accidental modification from outside functions
- ↳ It provides data hiding features

Analysis of OOP's / working of OOP

- OOP allows decomposition of a problem into no. of entities called objects and then builds data and functions around this objects
- Data of an object can be accessed only by the functions associated with that object.
- However functions of one object can access the functions of other objects.



C++

- It was developed by Bjarne straustup at bell laboratories in 1979
- since C++ is an attempt to add opp features in C, earlier it was called as 'c with objects'
- later in 1983 straustup named it as 'C++'

Advantages of C++

- 1) Portability
C++ provides this feature of portability by allowing us to develop codes without caring about the hardware.
- 2) mid-level programming language
It can treat a low level and high level language which help to develop games and desktop application
Whereas features of low level language helps to make

`iostream` / `conio.h` } libraries ← It consist some functions

Paradigm :- Systematic manner of writing something

Page No.	1	5
Date		

1) Kernels and drives

2) Object Oriented

The OOP concepts like polymorphism, encapsulation, inheritance and abstraction gives C++ the biggest advantage over other programming languages

3) Multi-paradigm Programming language

Paradigm refers to planning involved in programming. It is concern with the logic, the style and the way

how we proceed with the program. C++ is a multi-

paradigm programming language as it follows 3

Paradigm:

1) generic : using a single idea that serves multiple purpose

2) imperative : using steps that change the state of the program

3) object oriented : Using methods and class for reusability and modularity

4) wide range of application

C++ is useful to make GUI's as well as games

also it is useful to develop graphics and real time algebraic simulation. Hence C++ is beneficial in every stream

// ← used to comment a line
/*...*/ ← used to comment a paragraph

Page No.		
Date		

Syntax of C++

```
#include <iostream.h>
#include <conio.h>
using namespace std;
```

```
int main() { main function }
```

```
{
```

Object → cout << "Hello Fysal";
return 0; operator ?

6/12/22

Basic concepts of OOP's

→ Function

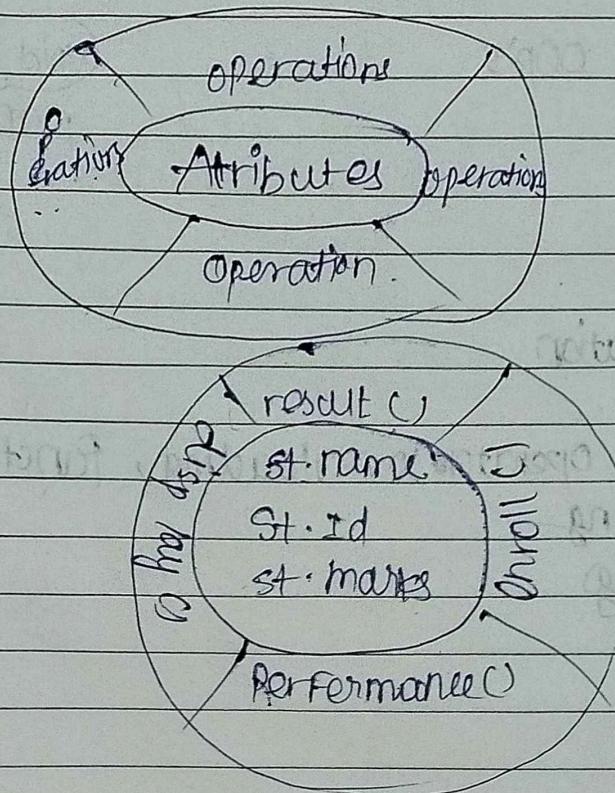
procedure call

- 1) Objects
- 2) class
- 3) Data abstraction
- 4) Data encapsulation
- 5) Inheritance
- 6) Polymorphism (operator overloading, function overloading)
- 7) Dynamic binding
- 8) Message passing

Basic concepts of oop's

Object

- Objects are the basic runtime entities in an object oriented system
- They maybe represented as a person, a place, a bank account, a table of data or 'any item' that the program must handle
- The fundamental idea behind object oriented approach is to combine both data and function into a single unit and this units are called objects
- The term objects means a combination of data and program that represent some real world entity

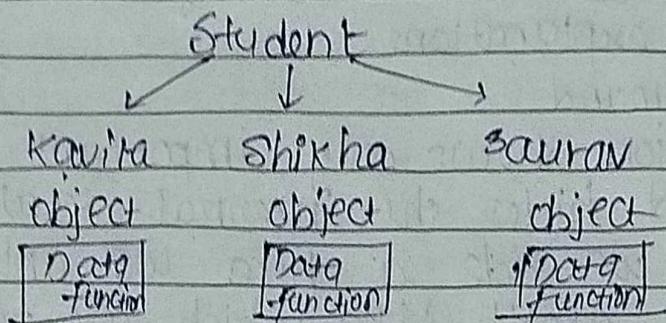


Class

- It is a group of object that share common properties for data part and some program part are collectively called a class

In C++ a class is a new data-type that contains member functions and member variables

Eg:-



Object

→ It is an instance of a class

Objects

→ In real world entity are pen, laptop, mobile

→ It is a physical entity

→ Object is created

through new keyword

Eg: Student s1 = new Student();

→ Objects allocate memory when it is created

→ It is a blueprint or template from which objects are created

→ It is a group of similar object.

→ It is a logical entity

→ Class is declared by using class keyword

Eg: Class A

→ Class doesn't allocate memory when it is created.

3) Data abstraction

- It refers to the act of representing essential features without including the background details or explanations
eg: Background
- Abstraction shows only important things to the user and hides the internal details
eg: When we ride a bike we only know some features about bike which are used for riding, but the internal details of functionality is hidden (not known by us)

4) Data encapsulation

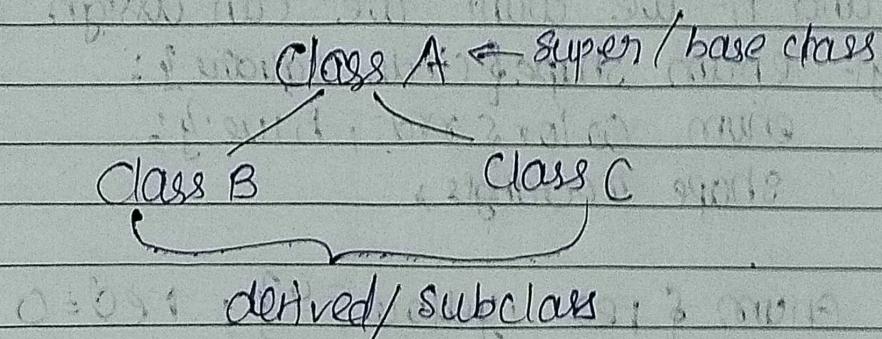
- The wrapping of data and functions into a single unit (called class) is known as encapsulation.
- The data is not accessible to the outside world and only those functions which are wrapped in the class can access it
- These functions provide interface between the objects data and program

eg:-

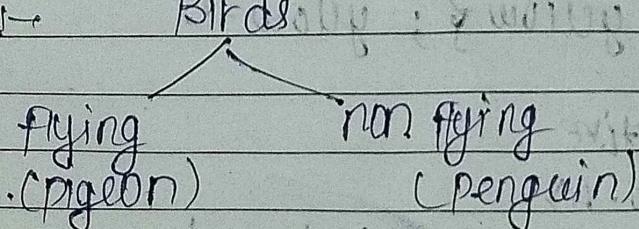
Class	Student
attribute	st.name, st.Id, st.marks
functions	result(), display() enroll(), performance()

5) Inheritance

- It is the process by which object of one class acquires the properties of another class.
- here, inheritance provides the idea of reusability. this means that we can add additional features to an existing class without modifying it.
- This is possible by designing a new class which will have the combined features of both the classes.
- The existing class is called as base class or super class and the new class is called as sub class or derived class.



eg :-



Application OOP's

- 1) Object oriented Database (OODBMS)
- 2) real time system design
- 3) simulation & modeling system
- 4) Hypertext and Hypermedia
- 5) AI expert system media
- 6) decision support and

(B) 12/22

Page No.	
Date	

Tokens

1) Reserved → Predefined

2) Identifiers → userdefined

3) Literals

→ sequence of characters

i) integer ii) character iii) float iv) string

4) Constant

keyword → const

eg:- const a = 10;

5) enum (enumeration)

eg:- enum (x, y, z);

x = 0, y = 1, z = 2 By default it starts from 0
and if we want we can assign value also

eg:- enum shape {circle, square};

enum color {red, blue};

shape rectangle;

enum {red, yellow} ; red = 0

enum {red = 4, yellow} ; yellow = 5

preprocessor or directive

eg:- #include <conio.h>

here semi colon(;) is not included

A) Operators

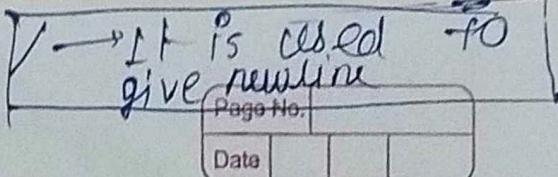
Special operators

1) :: scope variable number declaration

2) :: * pointer to word destination

3) - * pointer to word operator

4)



- 5) endl. line feed operator
- 6) new

- 7) :: scope variable

Scope resolution operator is used to recover a hidden value

- 8) Separators