OBJECT ORIENTED PROGRAMMING CONCEPTS

Object Oriented Programming follows bottom up approach in program design and emphasizes on safety and security of data..

FEATURES OF OBJECT ORIENTED PROGRAMMING:

Inheritance:

- Inheritance is the process of forming a new class from an existing class or base class. The base class is also known as parent class or super class.
- Derived class is also known as a child class or sub class. Inheritance helps in reusability of code , thus reducing the overall size of the program

Data Abstraction:

- It refers to the act of representing essential features without including the background details .Example : For driving , only accelerator, clutch and brake controls need to be learnt rather than working of engine and other details.

Data Encapsulation:

- It means wrapping up data and associated functions into one single unit called class..
- A class groups its members into three sections :public, private and protected, where private and protected members remain hidden from outside world and thereby helps in implementing data hiding.

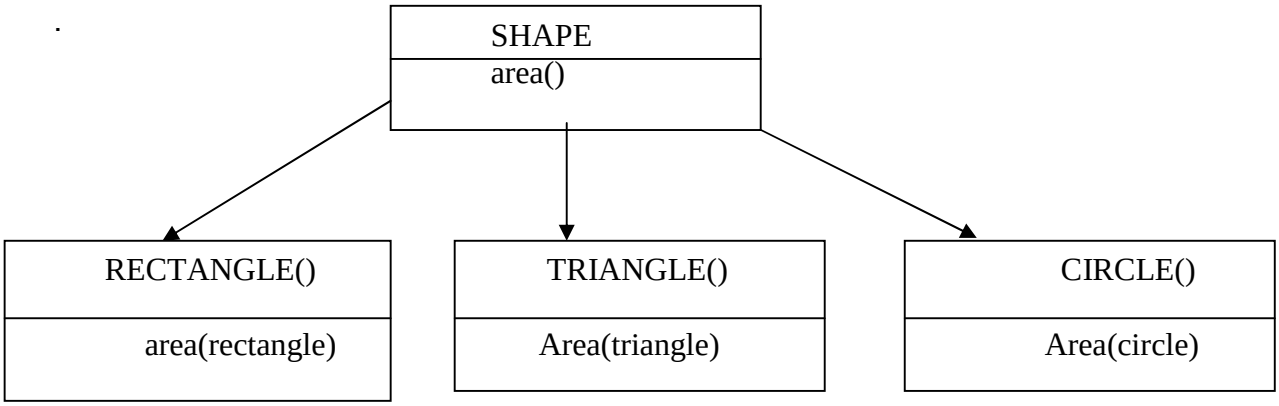
Modularity :

- The act of partitioning a complex program into simpler fragments called modules is called as modularity.
- It reduces the complexity to some degree and
- It creates a number of well defined boundaries within the program .

Polymorphism:

- **Poly** means many and **morphs** mean form, so polymorphism means one name multiple forms.
- It is the ability for a message or data to be processed in more than one form.
- C++ implements Polymorhism through Function Overloading , Operator overloading and Virtual functions .

.



Objects and Classes :

The major components of Object Oriented Programming are . **Classes & Objects**

A **Class i**s a group of similar objects . **Objects** share two characteristics: They all have *state* and *behavior*. For example : Dogs have state (name, color, breed, hungry) and behavior (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behavior (changing gear, applying brakes). Identifying the state and behavior for real-world objects is a great way to begin thinking in terms of object-oriented programming. These real-world observations all translate into the world of object-oriented programming.

Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in *fields* (variables in some programming languages) and exposes its behavior through functions

Classes in Programming :

- **It is a collection of variables, often of different types and its associated functions.**
- **Class just binds data and its associated functions under one unit there by enforcing encapsulation.**
- Classes define types of data structures and the functions that operate on those data structures.
- A class defines a blueprint for a data type.

Declaration/Definition :

A class definition starts with the keyword **class** followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations.

```
class class_name {
                        access_specifier_1:
                        member1;
                        access_specifier_2:
                        member2;
                        ...
                        } object_names;
```

Where class_name is a valid identifier for the class, object_names is an optional list of names for objects of this class. The body of the declaration can contain members that can either be data or function declarations, and optionally access specifiers.

[Note: the default access specifier is private.

Example : class Box { int a;

public:

double length; // Length of a box
double breadth; // Breadth of a box
double height; // Height of a box

};

Access specifiers in Classes:

Access specifiers are used to identify access rights for the data and member functions of the class. There are three main types of access specifiers in C++ programming language:

- private
- public
- protected

Member-Access Control

Type of Access	Meaning
Private	Class members declared as **private** can be used only by member functions and friends (classes or functions) of the class.
Protected	Class members declared as **protected** can be used by member functions and friends (classes or functions) of the class. Additionally, they can be used by classes derived from the class.
Public	Class members declared as **public** can be used by any function.

- **Importance of Access Specifiers**

  Access control helps prevent you from using objects in ways they were not intended to be used. Thus it helps in implementing data hiding and data abstraction.

OBJECTS in C++:

Objects represent instances of a class. Objects are basic run time entities in an object oriented system.

Creating object / defining the object of a class:

The general syntax of defining the object of a class is:-

Class_name object_name;

In C++, a class variable is known as an object. The declaration of an object is similar to that of a variable of any data type. The members of a class are accessed or referenced using object of a class.

```
Box Box1;          // Declare Box1 of type Box
Box Box2;          // Declare Box2 of type Box
```

Both of the objects Box1 and Box2 will have their own copy of data members.

Accessing / calling members of a class ***All member of a class are private by default***.
Private member can be accessed only by the function of the class itself.Public member of a class can be accessed through any object of the class. They are accessed or called using object of that class with the help of dot operator (.).

The general syntax for accessing data member of a class is:-

Object_name.Data_member=value;

The general syntax for accessing member function of a class is:-

Object_name. Function_name (actual arguments);

The dot ('. ') used above is called the **dot operator or class member access operator**. The dot operator is used to connect the object and the member function. The private data of a class can be accessed only through the member function of that class.

Class methods definitions (Defining the member functions)

Member functions can be defined in two places:-

- **Outside the class definition**

The member functions of a class can be defined outside the class definitions. It is only declared inside the class but defined outside the class. The general form of member function definition outside the class definition is:

Return_type Class_name:: function_name (argument list)
{
Function body
}

Where symbol :: is a scope resolution operator.

The scope resolution operator (::) specifies the class to which the member being declared belongs, granting exactly the same scope properties as if this function definition was directly included within the class definition

```
class sum
{
int A, B, Total;
public:
void getdata ();
void display ();
};
void sum:: getdata ()        // Function definition outside class definition Use of :: operator
{
cout<<" \n enter the value of A and B";
cin>>A>>B;
}
void sum:: display ()        // Function definition outside class definition Use of :: operator
{
Total =A+B;
cout<<"\n the sum of A and B="<<Total;
}
```

- **Inside the class definition**

The member function of a class can be declared and defined inside the class definition.

```
class sum
{
int A, B, Total;
public:
void getdata ()
{
cout< ”\n enter the value of A and B”;
cin>>A>>B;
}
void display ()
{
total = A+B;
cout<<”\n the sum of A and B=”<<total;
}
};
```

Differences between struct and classes in C++

In C++, a *structure* is a class defined with the `struct` keyword.Its members and base classes are public by default. A class defined with the `class` keyword has private members and base classes by default. This is the only difference between structs and classes in C++.

INLINE FUNCTIONS

- **Inline functions definition starts with keyword inline**
- **The compiler replaces the function call statement with the function code itself(expansion) and then compiles the entire code.**
- **They run little faster than normal functions as function calling overheads are saved.**
- **A function can be declared inline by placing the keyword inline before it.**

Example

```
inline  void Square (int a)

{ cout<<a*a;}

void main()

{.

 Square(4);        ⟶   { cout <<4*4;}

 Square(8)  ;      ⟶   { cout <<8*8; }

}
```

In place of function call , function body is substituted because Square () is inline function

Pass Object As An Argument

/*C++ PROGRAM TO PASS OBJECT AS AN ARGUMEMT. The program Adds the two heights given in feet and inches. */

```
#include< iostream.h>
#include< conio.h>
class height
{
int feet,inches;
public:
void getht(int f,int i)
{
feet=f;
inches=i;
}
void putheight()
{
cout< < "\nHeight is:"< < feet< < "feet\t"< < inches< < "inches"< < endl;
}
void sum(height a,height b)
{
height n;
n.feet = a.feet + b.feet;
n.inches = a.inches + b.inches;
if(n.inches ==12)
{
n.feet++;
n.inches = n.inches -12;
}
cout< < endl< < "Height is "< < n.feet< < " feet and "< < n.inches< < endl;
}};
void main()
{height h,d,a;
clrscr();
h.getht(6,5);
a.getht(2,7);
h.putheight();
a.putheight();
d.sum(h,a);
getch();
}
```

```
/**********OUTPUT***********
Height is:6feet 5inches
Height is:2feet 7inches
Height is 9 feet and 0
```

4 Marks Solved Problems :

Q 1) **Define a class TAXPAYER in C++ with following description :**

Private members :

- Name of type string
- PanNo of type string
- Taxabincm (Taxable income) of type float
- TotTax of type double
- A function CompTax( ) to calculate tax according to the following slab:

Taxable Income	**Tax%**
Up to 160000	0
>160000 and <=300000	5
>300000 and <=500000	10
>500000	15

Public members :

- A parameterized constructor to initialize all the members
- A function INTAX( ) to enter data for the tax payer and call function CompTax( ) to assign TotTax.
- A function OUTAX( ) to allow user to view the content of all the data members.

Ans.

```
class TAXPAYER
{
char Name[30],PanNo[30];
float Taxabincm;
double TotTax;
void CompTax()
{ if(Taxabincm >500000)
TotTax= Taxabincm*0.15;
else if(Taxabincm>300000)
TotTax= Taxabincm*0.1;
Else if(Taxabincm>160000)
TotTax= Taxabincm*0.05;
else
TotTax=0.0; }

public:
TAXPAYER(char nm[], char pan[], float tax, double tax) //parameterized constructor
{ strcpy(Name,nm);
strcpy(PanNo,pan);
Taxabincm=tax;
TotTax=ttax; }
void INTAX()
{ gets(Name);
cin>>PanNo>>Taxabincm;
CompTax(); }
void OUTAX()
{ cout<<Name<<'\n'<<PanNo<<'\n'<<Taxabincm<<'\n'<<TotTax<<endl; }
};
```

Q 2 : **Define a class HOTEL in C++ with the following description:**

Private Members

- Rno //Data Member to store Room No
- Name //Data Member to store customer Name
- Tariff //Data Member to store per day charge
- NOD //Data Member to store Number of days
- CALC //A function to calculate and return amount as NOD*Tariff and if the value of NOD*Tariff is more than 10000 then as 1.05*NOD*Tariff

Public Members:

- Checkin( ) //A function to enter the content RNo,Name, Tariff and NOD
- Checkout() //A function to display Rno, Name, Tariff, NOD and Amount (Amount to be displayed by calling function CALC( )

Solution :

```
#include<iostream.h>
class HOTEL
{       unsigned int Rno;
        char Name[25];
        unsigned int Tariff;
        unsigned int NOD;
        int CALC()
        {       int x;
                x=NOD*Tariff;
                if( x>10000)
                return(1.05*NOD*Tariff);
                else
                return(NOD*Tariff);
        }
        public:
        void  Checkin()
        {cin>>Rno>>Name>>Tariff>>NOD;}
        void Checkout()
        {cout<<Rno<<Name<<Tariff<<NOD<<CALC();}
        };
```

Q 3 Define a class Applicant in C++ with following description:

Private Members

- **A data member ANo (Admission Number) of type long**
- **A data member Name of type string**
- **A data member Agg(Aggregate Marks) of type float**
- **A data member Grade of type char**
- **A member function GradeMe( ) to find the Grade as per the Aggregate Marks obtained by a student. Equivalent Aggregate marks range and the respective Grades are shown as follows**

Aggregate Marks	Grade
> = 80	A
Less than 80 and > = 65	B
Less than 65 and > = 50	C
Less than 50	D

Public Members

- **A function Enter( ) to allow user to enter values for ANo, Name, Agg & call function GradeMe( ) to find the Grade**
- **A function Result ( ) to allow user to view the content of all the data members.**

Ans:
```
class Applicant
    {long  ANo;
    char Name[25];
    float Agg;
    char Grade;
    void GradeMe( )
    {       if (Agg > = 80)
                Grade = 'A';
            else if (Agg >= 65 && Agg < 80 )
                Grade = 'B';
            else if (Agg >= 50 && Agg < 65 )
                Grade = 'C;
            else
                Grade = 'D';
    }
public:
    void Enter ( )
    {       cout <<"\n  Enter Admission No.      ";   cin>>ANo;
            cout <<"\n  Enter Name of  the Applicant     "; cin.getline(Name,25);
            cout <<"\n  Enter Aggregate Marks obtained by the Candidate :";  cin>>Agg;
            GradeMe( );
    }
    void Result( )
    {       cout <<"\n  Admission No.    "<<ANo;
            cout <<"\n  Name of  the Applicant   ";<<Name;
            cout<<"\n  Aggregate Marks obtained by the Candidate.      " << Agg;
            cout<<\n    Grade Obtained  is    " << Grade ;
    }
};
```

Q 4 Define a class ITEM in C++ with following description:
Private members:

- **Icode of type integer (Item Code)**
- **Item of type string (Item Name)**
- **Price of type Float (Price of each item)**
- **Qty of type integer (Quantity in stock)**
- **Discount of type float (Discount percentage on the item)**
- **A find function finddisc( ) to calculate discount as per the following rule:**
  - **If Qty <=50 discount is 0%**
  - **If 50 < Qty <=100 discount is 5%**
  - **If Qty>100 discount is 10%**

Public members :
A function Buy( ) to allow user to enter values for Icode, Item,Price, Qty and call function Finddisc ( ) to calculate the discount.
A function showall ( ) to allow user to view the content of all the data members.

```
Ans : class ITEM
{int Icode,Qty;
char item[20];
float price,discount;
void finddisc();
public:
void buy();
void showall();
};
void stock::finddisc()
{If (qty<=50)
Discount=0;
Else if (qty> 50 && qty <=100)
Discount=0.05*price;
Else if (qty>100)
Discount=0.10*price;
}
void stock::buy()
{cout<<"Item Code :";cin>>Icode;
cout<<"Name :";gets(Item);
cout<<"Price :";cin>>Price;
cout<<"Quantity :";cin>>Qty;
finddisc();
}
void TEST::DISPTEST()
{cout<<"Item Code :";cout<<Icode;
cout<<"Name :";cout<<Item;
cout<<"Price :";cout<<Price;
cout<<"Quantity :";cout<<Qty;
cout<<"Discount :";cout<<discount;
}
```

4 marks Practice Problems :

Q 1 Define a class **employee** with the following specifications : **4**

Private members of class employee

- empno integer
- ename 20 characters
- basic, hra, da float
- netpay float
- calculate() A function to calculate basic + hra + da with float return type

Public member function of class employee

- havedata() function to accept values for empno, sname, basic, hra, da and invoke calculate() to calculate netpay.
- dispdata() function to display all the data members on the screen.

Q2 Define a class **Student** with the following specifications : **4**

Private members :

- roll_no integer
- name 20 characters
- class 8 characters
- marks[5] integer
- percentage float

- Calculate() a function that calculates overall percentage of marks and return the percentage of marks.

public members :

- Readmarks() a function that reads marks and invoke the Calculate function.
- Displaymarks() a function that prints the marks.

Q3 : Define a class **repor**t with the following specification : **4**

Private members :

- adno 4 digit admission number
- name 20 characters
- marks an array of 5 floating point values
- average average marks obtained
- getavg() to compute the average obtained in five subjects

Public members :

- readinfo() function to accept values for adno, name, marks, and invoke the function getavg().
- displayinfo() function to display all data members on the screen you should give function definitions.

Q4 Declare a class **myfolder** with the following specification : **4**

Private members of the class

- Filenames – an array of strings of size[10][25](to represent all the names of files inside myfolder)
- Availspace – long (to represent total number of bytes available in myfolder)
- Usedspace – long (to represent total number of bytes used in myfolder)

public members of the class

- Newfileentry() – A function to accept values of Filenames, Availspace and Usedspace fromuser
- Retavailspace() – A Fucntion that returns the value of total Kilobytes available (1 Kilobytes = 1024 bytes)
- Showfiles() – a function that displays the names of all the files in myfolder

2 Marks Practice Problems

1. What is relation between class and object?
2. What are inline functions? Give example
3. Difference between private & public access specifiers.
4. How class implements data-hiding & encapsulation?
5. What is the difference between structure and a class ?
6. How is inline function different from a normal function ?