4.CLASSES AND OBJECTS

2010 Delhi:

1.(c) Rewrite the following c++ program code after removing the syntax error(s) (if any). Underline each correction. 2

```
include <iostream.h>
class TRAIN
{
long TrainNo;
char Description[25];
public
void Entry ( )
{
cin >>TrainNo; gets(Description);
}
Void Display ( )
{
cout<<TrainNo<<":"<<Description<<endl;
}
};
void main( )
{
TRAIN T;
Entry. T( ); Display. T( );
}
```

Ans.

```
 #include<iostream.h>
#include<stdio.h>
class TRAIN
{
        long TrainNo;
        char Description [25];
    public:
        void Entry ()
        {
         cin>>TrainNo; gets (Description);
        }
        void Display ()
        {
        cout<<TrainNo<<":"<<Description<<end1;
        }
};
void main ()
{
        TRAIN T;
        T.Entry();
         T.Display();
}
```

2.c) Define a class ITEM in C++ with following description: 4

Private Members
_ Code of type integer (Item Code)
_ Iname of type string (Item Name)
_ Price of type float (Price of each item)
_ Qty of type integer (Quantity of item in stock)
_ Offer of type float (Offer percentage on the item)
_ A member function GetOffer() to calculate Offer percentage as per the following rule:
If Qty<=50 Offer is 0
If 50<Qty<=100 Offer is 5
If Qty>100 Offer is 10

Public Members
_ A function GetStock() to allow user to enter values for Code, Iname, Price, Qty and call function GetOffer() to calculate the offer
_ A function ShowItem() to allow user to view the content of all the data members

Ans.

```
class ITEM
{
        int Code;
        char Iname [20] ;
        float Price;
        int Qty;
        float Offer;
        void GetOffer() ;
    public:
        void GetStock ()
        {
        cin>>Code;
        gets (Iname) ;
     // OR cin.getline (Iname, 80) ; OR cin>>Iname;
        cin>>Price>>Qty;
        GetOffer() ;
}
void ShowItern ( )
{
        cout<<Code<<Iname<<Price<<Qty<<Offer
;
};
void ITEM: : GetOffer ()
{
        if (Qty<=50)
                Offer = 0;
        else if (Qty <=100)
                Offer = 5; / /OR Offer = 0.05;
        else
                Offer = 10; / /OR Offer = 0.1;
}
```

2010 Outside Delhi:

1. (c) Rewrite the following C++ program code after removing the syntax error(s) (if any). Underline each correction. 2

```
include <iostream.h>
class FLIGHT
{
long FlightCode;
char Description[25];
```

```
public
void AddInfo ( )
{
cin>>FlightCode; gets (Description) ;
{
void ShowInfo ()
(
cout<<FlightCode<<":"<<Description<<endl;
}
} ;
void main()
{
FLIGHT F;
AddInfo.F(); ShowInfo.F();
}
```

Ans.

```
 #include <iostream.h> / / Error 1
#include <stdio.h>        / / Error 2
class FLIGHT
{
  long FlightCode;
  //not required if gets( ) is re-placed with
  //cin.getline( ) or cin
  char Description[25];
public :                        / / Error 3
   void AddInfo ( )
   {
      cin>>FlightCode; gets (Description) ;
    }
    void ShowInfo ( )
    {
      cout<<FlightCode<<":"<<Description<<e
ndl;
     }
} ;
void main ( )
{
     FLIGHT F;
     F.AddInfo( ) ;
     F. ShowInfo ( ) ;              / / Error 4
}
```

2(c) Define a class STOCK in C++ with following description: 4

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 member 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 STOCK
{          int ICode,Qty;
           char Item[20];
           float Price,Discount;
           void FindDisc();
    public:
           void Buy();
           void ShowAll();
} ;
void STOCK::Buy()
{          cin>>ICode;
           gets(Item);
           cin>>Price;
           cin»Qty;
           FindDisc();
}
void STOCK::FindDisc()
{          if (Qty<=50)
                    Discount=0;
           else if (Qty<=100)
                     Discount=5; // =0.05;
           Else
                     Discount=10; // =0.1;
}
void STOCK::ShowAll()
{
cout<<ICode<<'\t'<<Item<<'\t'<<Price<<'\t'<<Qty
              <<'\t'<<Discount<<endl;
}
```

2009 Delhi:

(c) Define a class RESORT in C++ with following description: 4

Private Members

_ Rno *//Data* member to store Room No
_ Name *//Data* member to store customer name
_ Charges *//Data* member to store per day charges
_ Days *//Data* member to store number of days of stay
_ COMPUTE( ) //A function to calculate' and return Amount as Days*Charges and if the value of Days*Charges is more than 11000 then as 1.02*Days*Charges

Public Members

_ Getinfo ( ) *//A* function to enter the content Rno, Name ,*Charges* and Days
_ Dispinfo ( ) *//A* function to display Rno, Name, Charges,*Days* and Amount (Amount to be displayed by *calling* function COMPUTE ( ))

Ans

```
 class RESORT
{         int Rno;
          char Name [20];
          float Charges;
          int Days;
          float COMPUTE();
       public:
          void Getinfo() ;
          void Dispinfo();
};
void RESORT::Getinfo()
{         cin>>Rno;
          gets (Name);
          cin>>Charges;
          cin>>Days;
}
void RESORT::Dispinfo()
{
cout<<Rno<<" "<<Name<<" "<<Charges<<"
              "<<Days<< COMPUTE()<<endl;
}
float RESORT::COMPUTE(}
{
          float Amount = Charges*Days;
          if (Amount>11000)
                Amount = 1.02*Days*Charges;
          return Amount;
}
```

2009 Outside Delhi:

1.(c) Rewrite the following program after removing the syntactical errors (if any). Underline each correction. 2

```
include <iostream.h>
include <stdio.h>
class MyStudent
{
int StudentId = 1001;
char Name [20] ;
public
MyStudent( ){ }
void Register ( ) {cin>>StudentId; gets (Name) ;}
void Display ( ) {cout<<StudentId<< ":"
<<Name<<end1;}
} ;
void main ( )
{
MyStudent MS ;
Register.MS( ) ;
MS.Display( ) ;
}
```

Ans

```
 # include <iostream.h>
# include <stdio.h>
class MyStudent
{     int StudentId;
          char Name[20];
      public :
          MyStudent( )
          {
              StudentId = 1001;
          }
            void Register( )
            {
                cin>>StudentId;
                gets (Name);
            }
          void Display ()
          {
           cout«StudentId<<":"<<Name<<endl;
          }
};
void main ()
{         MyStudent MS;
          MS. Register ();
          MS. Display () ;
}
```

2. (c) Define a class HOTEL in C++ with the following description: 4

Private Members:

_ Rno //Data member to store *Room* No
_ Name //Data member to store customer name
_ Tariff //Data member to store per day charges
_ NOD //Data member to store number of days of stay
_ 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( ))

Ans

```
class HOTEL
{         int Rno;
          char Name[20];
          float Tariff;
          int NOD;
          float CALC() ;
    public:
          void Checkin() ;
          void Checkout() ;
} ;
float HOTEL::CALC()
{    float Amount = Tariff*NOD;
          if (Amount>10000)
                  Amount = 1.05*NOD*Tariff;
          return Amount;
```

```
}
void HOTEL::Checkin()
{
        cin>>Rno;
        gets (Name);
        cin>>Tariff;
        cin>>NOD;
}
void HOTEL::Checkout()
{
      cout<<Rno<<" "<<Name<<" "<<Tariff<<"
                    "<<NOD<<CALC ()<<endl;
}
```

1.c) Rewrite the following program after removing the syntactical errors (if any). Underline each correction. 2

```
include <iostream.h>
include <stdio.h>
class MyStudent
{  int StudentId=1001;
  char Name[20];
 public
  MyStudent( ) { }
  void Register( )
  {   cin>>StudentId;
      gets(Name);
  }
  void Display( )
  {   cout<<StudentId<<":"<<Name<<endl;
  }
};
void main( )
{   MyStudent MS;
   Register.MS( );
   MS.Display( );
}
```

Ans:

```
#include <iostream.h>
#include <stdio.h>
class MyStudent
{  int StudentId;
  char Name[20];
 public:
  MyStudent( ) { }
  void Register( )
  {   cin>>StudentId;
      gets(Name);
  }
  void Display( )
  {   cout<<StudentId<<":"<<Name<<endl;
  }
};
void main( )
{   MyStudent MS;
   MS.Register( );
   MS.Display( );
}
```

2008 Delhi:

2.a) Differentiate between public and private visibility modes in context of Object Oriented Programming using a suitable example illustrating each.

Ans: public and private visibility modes in context of OOP:

The visibility mode (private or public or protected) in the definition of the derived class specifies whether the features of the base class are privately derived or publicly derived or protected derived. The visibility modes basically control the access specifier to be for inheritable members of base class, in the derived class.

Public visibility mode: The public derivation means that the derived class can access the public and protected members of the base class but not the private members of the base class. With publicly derived class, the public members of the base class become the public members of the derived class, and the protected members of the base class become the protected members of the derived class.

Private visibility mode: The private derivation means, the derived class can access the public and private members of the base class privately. With privately derived class, the public and protected members of the base class become private members of the derived class. That means the inherited members can be accessed only through member functions of the derived class.

Visibility Mode	Inheritable **public** member becomes (in derived class)	Inheritable **protected** member becomes (in derived class)	Private member of base class are not directly accessible to derived class.
public	Public	protected	
privatee	Private	private	

public and private access specifiers in context of OOP: public access specifier is used to define any method or a variable which may be accessed by any member function of the same class and also from outside the class. Private access specifier is used to make any variable or a method which has a limited access within the class only.The concept of data hiding is implemented through the private access specifier only.

Eg:

```
class student
{   private:
            int rno;
            char name[21];
   public:
             int age;
             void input( );
             void display( );
}
```

Here, since rno and name are declared in private, they can be accessed only inside the class. Since age,input( ) and display() are declared in public, they can be accessed from outside class also.

2008 Outside Delhi:

2.a) Differentiate between private and protected visibility modes in context of object oriented programming using a suitable example illustrating each.

Ans: private and protected visibility modes in context of OOP:

The visibility mode (private or public or protected) in the definition of the derived class specifies whether the features of the base class are privately derived or publicly derived or protected derived. The visibility modes basically control the access specifier to be for inheritable members of base class, in the derived class.

Private visibility mode: The private derivation means, the derived class can access the public and private members of the base class privately. With privately derived class, the public and protected members of the base class become private members of the derived class. That means the inherited members can be accessed only through member functions of the derived class.

Protected visibility mode: The protected derivation means that the derived class can access the public and private members of the base class protectedly. With protectedly derived class, the public and protected members of the base calss become protected members of the derived class. That means the inherited members are now not available to the outside world and can be accessed only through the member functions of the derived class and the classes based upon the derived classes. These members can be inherited further if any classes are inheriting from the derived class.

Visibility Mode	Inheritable **public** member becomes (in derived class)	Inheritable **protected** member becomes (in derived class)	Private member of base class are not directly accessible to derived class.
protected	Protected	protected	
private	Private	private	

private and protected access specifiers in context of OOP:

private access specifier is used to make any variable or a method which has a limited access within the class only.

At the time of inheritance, these variables cannot be accessed (inherited) to the derived class.

protected access specifier is used to make any variable or a method which has a limited access within the class only (here like private). But at the time of inheritance, these variables can be inherited to the derived class.

Except regarding inheritance, both access specifiers ie private and protected will work same.

Eg:

```
class student
{   private:
            int rno;
            char name[21];
   protected:
             int age;
             void input( );
             void display( );
}
```

Here, since rno and name are declared in private, they can be accessed only inside the class. Since age,input( ) and display() are declared in protected, they also can be accessed only inside the class but they can be inherited, where as private members (rno and name) cannot be inherited.

2006 Delhi:

2.c) Define a class named ADMISSION in C++ with the following descriptions:

Private Members:

AD_NO	integer(Ranges 10 – 2000)
NAME	Array of characters(String)
CLASS	Character
FEES	Float

Public Members:

Function Read_Data( ) to read an object of ADMISSION type. Function Display( ) to display the details of an object. Function

Draw-Nos.( ) to choose 2 students randomly. And display the details. Use random function to generate admission nos. to match with AD_NO.

Ans:

```
class ADMISSION
{   int AD_NO;
   char NAME[31];
   char CLASS;
   float FEES;
public:
   void Read_Data( )
   { cout<<"\nEnter the Admission Number: ";
     cin>>AD_NO;
     cout<<"\nEnter the Student Name: ";
     gets(NAME);
     cout<<"\nEnter the Class: ";
     cin>>CLASS;
     cout<<"\nEnter the Fees: ";
     cin>>FEES;
   }
   void Display()
   { cout<<"\nThe Admission Number of the
            student: "<<AD_NO;
     cout<<"\nThe name of the Student: "
            <<NAME;
     cout<<"\nThe Class of the Student:"
            <<CLASS;
     cout<<"\nThe Fees of the Student: "
            <<FEES;
   }
   void Draw_Nos();
};
void ADMISSION::Draw_Nos( )
{    //Dear Students, a test for you. Complete
this member function.

}
```

2006 Outside Delhi:

1.b) Illustrate the use of Inline function in C++ with the help of an example. 2

Ans: INLINE FUNCTIONS: The inline functions are a C++ enhancement designed to speed up programs. The coding of normal functions and inline functions is similar except that inline functions definitions start with the keyword inline.

The working of inline functions:

After writing any program, it is first compiled to get an executable code, which consists of a set of machine language instructions. When this executable code is executed, the operating system loads these instructions into the computer's memory, so that each instruction is stored in a specific memory location. Thus, each instruction has a particular memory address.

After loading the executable program in the computer memory, these instructions are executed step by step. When a function call instruction is encountered, the program stores the memory address of the instruction immediately following the function call statement, loads the function being called into the memory, copies argument values, jumps to the memory location of the called function, executes the function code, stores the return value of the function, and then jumps back to the address of the instruction that was saved just before executing the called function.

With inline code, the compiler replaces the function call statement with the function code itself (this process is called expansion) and then compiles the entire code. Thus, with inline functions, the compiler does not have to jump to another location to execute the function, and then jump back as the code of the called function is already available to the calling program.

Inline functions run a little faster than the normal functions as function calling overheads are saved, however there is a memory penalty. If 10 times an inline function is called, there will be 10 copies of the function inserted into the code.

A function can be declared inline by placing the keyword inline before it. An inline function definition should be placed above all the functions that call it. The functions should be inlined only when they are small. Since for large functions, they will become memory penalty.

The inlining does not work for following situations:

a. For functions that return values and are having a loop or a switch or a goto.
b. For functions not returning values, if a return statement exists.
c. If functions contain static variables.
d. If the function is recursive(a function that calls itself).

Inlining and the member functions:

The member function of a class, if defined within the class definition, are inlined by default. Therefore, only very small member functions should be defined within the class definition.

The member functions defined outside the class definition can be made explicitly inline by placing the keyword inline before their definition.

Inline functions are best for small functions that are called often.The compiler may even ignore

your attempt to linline a function if it consists more than 50 lines of code.

2. c) Define a class named HOUSING in C++ with the following descriptions: 4

Private Members:

REG_NO	integer(Ranges 10-1000)
NAME	Array of characters(String)
TYPE	Character
COST	Float

Public Members:

Function Read_Data( ) to rread an object of HOUSING type.
Function Display( ) to display the details of an object.
Function Draw_Nos( ) to choose and display the details of 2 houses selected randomly from an array of 10 objects of type HOUSING. Use random function to generate the registration nos. to match with REG_NO from the array.

Ans:

```
class HOUSING
{  int REG_NO;
   char NAME[31];
   char TYPE;
   float COST;
public:
   void Read_Data( )
   { cout<<"\nEnter the House Registration
           Number: ";
     cin>>REG_NO;
     cout<<"\nEnter the House Name: ";
     gets(NAME);
     cout<<"\nEnter the House Type: ";
     cin>>TYPE;
     cout<<"\nEnter the House Cost: ";
     cin>>COST;
   }
   void Display()
   { cout<<"\nThe Registration Number of the
            House: "<<REG_NO;
     cout<<"\nThe name of the House: “
            <<NAME;
     cout<<"\nThe Type of the House: "<<TYPE;
     cout<<"\nThe Cost of the House: "<<COST;
   }
   void Draw_Nos();
};
void HOUSING::Draw_Nos( )
{ //Dear Students, a test for you.  Complete this
member function.

}
```

2004 :

2.b) Declare a class myfolder with the following specifications:

Private members of the class:

Filenames an array of strig 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 from user.
Retavailspace(): A function that returns the value of total kilobytes available
(1 kilobyte=1024 bytes)
Showfiles( ): A function that displays the names of all the files in myfolder

Ans:

```
class myfolder
{  char Filenames[10][25];
  long Availspace;
  long Usedspace;
public:
  void Newfileentry( )
  {
      cout<<"\nEnter any 10 file names: ";
      for(int i=0;i<=9;i++)
     {cout<<"\nEnter the "<<i+1<<" file name:
";
      gets(Filenames[i]);
      }
cout<<"\nEnter the Available Space (In
              Kilobytes): ";
 cin>>Availspace;
 cout<<"\nEnter the Used Space (In
             Kilobytes): ";
cin>>Usedspace;
  }
  long RetavailSpace( )
  {     ret Availspace;
  }
  void Showfiles( )
  {  cout<<"\nThe names of the files in
            myfolder object....";
    for(i=0;i<=9;i++)
      {     puts(Filenames[i]);
            cout<<endl;
      }
  }
```

2002:

2.a) What do you understand about a member function? How does a member function differ from an ordinary function?

Ans: A member function is a function declared within a class. It is said to be defined in two ways. Ie Outside the class and inside the class. When a member function is defined outside the class, the name of the function must be the full name including the class name as well. When a member function is defined inside the class, the name of the function is similar to an ordinary function but it will become an **inline** function.

2.b) Define a class Student for the following specifications.
Private members of the Student are:
roll_no integer
name array of characters of size 20
class_st array of characters of size 8
marks array of integers of size 5
Percentage float
Calculate( ) that calculates overall percentage marks and returns the percentage
Public Members of the Student are:
Readmarks reads mark and invoke the calculate function
Displaymarks prints the data.

Ans:

```
class Student
{   int roll_no;
    char name[20];
    char class_st[8];
    int marks[5];
    float percentage;
    float calculate( )
    { percentage=(marks[0]+marks[1]+marks[2]
+
          marks[3]+marks[4])/5;
      return percentage;
    }
  public:
    void Readmarks( )
    { cout<<"\nEnter any 5 subject marks;
      cin>>marks[0]>>marks[1]>>marks[2]>>
                marks[3]>>marks[4];
       calculate( );
     }
     void Displaymarks( )
     { cout<<"\nThe Roll Number of the
                Student: "<<roll_no;
       cout<<"\nThe Name of the Student:"
                <<name;
       cout<<"\nThe class of the Student:  "
                <<class_st;
       cout<<"\n5 subject marks of the
                student…\n";
       cout<<marks[0]<<"\t"<<marks[1]<<"\t"<<
                marks[2]<<"\t";
       cout<<marks[3]<<"\t"<<marks[4]<<"\n";
       cout<<"Percentage ="<<percentage;
     }
};
```

2001:

2.b) Declare a class to represent bank account of 10 customers with the following data members. Name of the depositor, account number, type of account (S for Savings and C for Current), Balance amount. The class also contains member functions to do the following:
(i)To initialize data members.
(ii) To deposit money
(iii)To withdraw money after checking the balance (minimum balance is Rs.1000)
(iv) To display the data members.
[Note:You are also required to give detailed function definitions.]

```
class Bank
{       char name[15];
      int acc_no;
      char acc_type;
      float bal_amount;
   public:
      void readData( )
      {    cout<<"\nEnter the name: ";
           gets(name);
          cout<<"\nEnter the account number: ";
           cin>>acc_no;
           cout<<"\nEnter the account type: ";
           cin>>acc_type;
           cout<<"\nEnter the amount to deposit: ";
           cin>>bal_amount;
      }
      void deposit( )
      { float deposit;
       cout<<"\nEnter your account number: ";
       cin>>acc_no;
       cout<<"\nEnter the amount to deposit: ";
       cin>>deposit;
       bal_amount=bal_amount + deposit;
      }
      void withdraw( )
      {  float w_amount;
        cout<<"\nEnter your account number: ";
         cin>>acc_no;
         cout<<"\nEnter amount to withdraw";
         cin>>w_amount;
         if((bal_amount-w_amount)<1000)
             cout<<"\nWithdraw is not possible";
         else
         {  bal_amount=bal_amount-w_amount;
            cout<<"\nThe balance is
                  "<<bal_amount-w_amount;
         }
      }
      void display( )
      {   cout<<"\nName of the depositor: "
             <<name;
        cout<<"\nAccount Number: "<<acc_no;
```

```
        cout<<"\nAccount Type: "<<acc_type;
        cout<<"\nThe balance amount is
            "<<bal_amount;
    }
};
```

2000 :

2.b) Define a class worker with the following specification. 4

Private member of class worker:

wname 25characters
hrwrk,wgrate float (hours worked and wagerate per hour)
totwage float(hrwrk*wgrate)
cakcwg() A function to find hrwrk*wgrate with float return type

Public members of class worker:

In_data( ): A function to accept values for wno, wname, hrrwrk, wgrate and invoke calcwg( ) to calculate totpay.
Out_data( ): A function to display all the data members on the screen you should give definitions of functions.

```
class worker
{   char wname[25];
    float hrwrk,wgrate;
    float totwage;
    float cakcwg( )
    {  return hrwrk*wgrate;
    }
public:
   void In_data( )
   {  cout<<"\nEnter Worker number,name,
          hours worked and wage rate";
      cin>>wno;
      gets(wname);
      cin>>hrwrk>>wgrate;
      calcwg( );
   }
 void Out_data( )
 { cout<<"\nThe Worker Number: "<<wno;
   cout<<"\nThe Name of the worker:
          "<<wname;
   cout<<"\nNumber of hours worked by the
          worker: "<<hrwrk;
   cout<<"\nThe Wage Rate of the Worker:
         "<<wgrate;
   cout<<"\nThe total wages of the worker:
        "<<totwage;
 }
```

1999 :

2.b) Define a class Teacher with the following class specification:

Private members:

Name 20 characters
Subject 10 characters
Basic, DA, HRA float
Salary float
Calculate( ) function computes the salary and returns it. Salary is sum of Basic, DA and HRA

Public members:

ReadData( ): Function accepts the data values and invoke the calculate function.
DisplayData( ):Function prints the data on the screen.

```
class Teacher
 {   char Name[20];
     char subject[10];
     float Basic,DA,HRA,Salary;
     float Calculate( )
     {     Salary=Basic+DA+HRA;
           return Salary;
     }
     public:
     void ReadData( )
     {    cout<<"\nEnter Basic, Dearness
                 Allowance and "
          cout<<" House Rent Allowance: ";
          cin>>Basic>>DA>>HRA;
          Calculate();
     }
     void DisplayData( )
     {     cout<<"\nThe Basic : "<<Basic;
           cout<<"\nThe Dearness
                    Allowance: "<<DA;
          cout<<"\nThe House Rent
                   Allowance: "<<HRA;
          cout<<"\nThe Salary: "<<Salary;
     }
 };
```

1998:

2.b) Define a class student with the following specifications:

Private members of class student:

Admno integer
Sname 20 character
English float
Math float
Science float
Total float
Ctotal( ) A function to calculate English + math + science with float return type

Public member functions of class student:

Takedata( ):Function to accept values for admno,sname, English, math, science and invoke ctotal to calculate total.
Showdata( ):Function to display all the data members on the screen.

```
class student
{    int Admno;
     char Sname[20];
          float English,Math,Science,Total;
          float Ctotal()
          {   Total=English+math+science;
              return Total;
          }
          public:
          void Takedata()
          {  cout<<"\nEnter the admission
                number,name of the student: ";
             cin>>Admno;
             gets(sname);
             cout<<"\nEnter English, Maths,
                              Science Marks: ";
              cin>>English>>Math>>Science;
              Ctotal( );
          }
          void Showdata( )
          {    cout<<"\nThe admission number of
                     the student: "<<Admno;
               cout<<"\nThe name of the student:
                                  "<<Sname;
               cout<<"\nEnglish , Maths and
                     Science Marks are…";
               cout<<english<<"\t"<<math<<"\t"
                           <<science<<"\n";
               cout<<"\nTotal marks of the
                           student: "<<Total;
     };
```

Model Paper 1 for 2008-09 Batch:

b)Rewrite the following program after removing the syntactical errors (if any). Underline each correction. 2

```
#include [iostream.h]
class PAYITNOW
{
          int Charge;
        PUBLIC:
          void Raise(){cin>>Charge;}
          void Show{cout<<Charge;}
        };
        void main()
        {
            PAYITNOW P;
            P.Raise();
            Show();
        }
```

Answer:

```
#include <iostream.h>
class PAYITNOW
{    int Charge;
public:
   void Raise(){cin>>Charge;}
   void Show(){cout<<Charge;}
};
```

```
void main()
{    PAYITNOW P;
     P.Raise();
     P.Show();
}
```

Model Paper 1 for 2008-09 Batch:

2.e) Define a class TEST in C++ with following description: 4

Private Members

a. TestCode of type integer
b. Description of type string
c. NoCandidate of type integer
d. CenterReqd (number of centers required) of type integer
e. A member function CALCNTR() to calculate and return the number of centers as (NoCandidates/100+1)

Public Members

- A function SCHEDULE() to allow user to enter values for TestCode, Description, NoCandidate & call function CALCNTR() to calculate the number of Centres
- A function DISPTEST() to allow user to view the content of all the data members

Answer:

```
class TEST
{         int TestCode;
          char Description[20];
          int NoCandidate,CenterReqd;
          void CALCNTR();
public:
          void SCHEDULE();
          void DISPTEST();
};
void TEST::CALCNTR()
{         CenterReqd=NoCandidate/100
+ 1;
}
void TEST::SCHEDULE()
{cout<<"Test Code   :";
 cin>>TestCode;
cout<<"Description :";
gets(Description);
cout<<"Number      :";
cin>>NoCandidate;
  CALCNTR();
}
void TEST::DISPTEST()
{
cout<<"Test Code   :"<<TestCode<<endl;
cout<<"Description :"<<Description<<endl
;
cout<<"Number      :"<<NoCandidate<<end
l;;
```

```
cout<<"Centres    :"<<CenterReqd<<endl;;
}
```

```
cout<<"Distance   :"<<Distance<<endl;;
cout<<"Fuel       :"<<Fuel<<endl;;
}
```

Model Paper 2 for 2008-09 Batch:

2.d) Define a class in C++ with following description: 4

Private Members

*A data member Flight number of type integer
*A data member Destination of type string
*A data member Distance of type float
*A data member Fuel of type float
*A member function CALFUEL() to calculate the value of Fuel as per the following criteria

Distance	Fuel
<=1000	500
more than 1000 and <=2000	1100
more than 2000	2200

Public Members

*A function FEEDINFO() to allow user to enter values for Flight Number, Destination, Distance & call function CALFUEL() to calculate the quantity of Fuel
*A function SHOWINFO() to allow user to view the content of all the data members

Answer:

```
class FLIGHT
{       int Fno;
        char Destination[20];
        float Distance, Fuel;
        void CALFUEL();
public:
        void FEEDINFO();
        void SHOWINFO();
};
void FLIGHT::CALFUEL()
{       if (Distance<1000)
                Fuel=500;
        else
                if (Distance<2000)
                        Fuel=1100;
                else
                        Fuel=2200;
}
void FLIGHT::FEEDINFO()
{cout<<"Flight No   :";cin>>Fno;
cout<<"Destination :";gets(Destination);
cout<<"Distance    :";cin>>Distance;
CALFUEL();
}
void FLIGHT::SHOWINFO()
{
cout<<"Flight No   :"<<Fno<<endl;
cout<<"Destination :"<<Destination<<endl;
```

Sample Paper 1 for 2009-10 Batch:

1.C) Rewrite the following program after removing the syntactical errors (if any). Underline each correction. 2

```
#include [iostream.h]
class MEMBER
{
int Mno;float Fees;
PUBLIC:
void Register(){cin>>Mno>>Fees;}
void Display{cout<<Mno<<" :
"<<Fees<<endl;}
};
void main()
{
MEMBER M;
Register();
M.Display();
}
```

A)

```
#include <iostream.h>
class MEMBER
{   int Mno;float Fees;
   public:
        void Register()
        {       cin>>Mno>>Fees;
        }
        void Display()
         {       cout<<Mno<<":"<<Fees<<endl;
        }
};
void main()
{       MEMBER M;
        M.Register();
        M.Display();
}
```

2.c) Define a class TEST in C++ with following description: 4

Private Members

• TestCode of type integer
• Description of type string
• NoCandidate of type integer
• CenterReqd (number of centers required) of type integer
• A member function CALCNTR() to calculate and return the number of centers as (NoCandidates/100+1)

Public Members

• A function SCHEDULE() to allow user to enter values for TestCode,
Description, NoCandidate & call function CALCNTR() to calculate the number of Centres
• A function DISPTEST() to allow user to view the content of all the data members

A)

```
class TEST
{
        int TestCode;
        char Description[20];
        int NoCandidate,CenterReqd;
        void CALCNTR();
   public:
        void SCHEDULE();
        void DISPTEST();
};
void TEST::CALCNTR()
{
    CenterReqd=NoCandidate/100 + 1;
}
void TEST::SCHEDULE()
{
   cout<<"Test Code :";cin>>TestCode;
   cout<<"Description :";gets(Description);
   cout<<"Number :";cin>>NoCandidate;
    CALCNTR();
}
{
   cout<<"Test Code :"<<TestCode<<endl;
   cout<<"Description :"<<Description<<endl;
   cout<<"Number :"<<NoCandidate<<endl;;
   cout<<"Centres :"<<CenterReqd<<endl;;
}
```

Sample Paper 2 for 2009-10 Batch:

2.c) Define a class in C++ with following description: 4

Private Members
• A data member Flight number of type integer
• A data member Destination of type string
• A data member Distance of type float
• A data member Fuel of type float
• A member function CALFUEL() to calculate the value of Fuel as per the following criteria

Distance	Fuel
<=1000	500
more than 1000 and <=2000	1100
more than 2000	2200

Public Members
" A function FEEDINFO() to allow user to enter values for Flight Number,
Destination, Distance & call function CALFUEL() to calculate the quantity of Fuel
" A function SHOWINFO() to allow user to view the content of all the data members

A)

```
class FLIGHT
{       int Fno;
        char Destination[20];
        float Distance, Fuel;
        void CALFUEL();
   public:
        void FEEDINFO();
        void SHOWINFO();
};
void FLIGHT::CALFUEL()
{
        if (Distance<=1000)
                Fuel=500;
        else
        if (Distance<=2000)
                Fuel=1100;
        else
        Fuel=2200;
}
void FLIGHT::FEEDINFO()
{
        cout<<"Flight No :";cin>>Fno;
      cout<<"Destination :";gets(Destination);
      cout<<"Distance :";cin>>Distance;
      CALFUEL();
}
void FLIGHT::SHOWINFO()
{
      cout<<"Flight No :"<<Fno<<endl;
      cout<<"Destination :"<<Destination<<endl;
      cout<<"Distance :"<<Distance<<endl;;
      cout<<"Fuel :"<<Fuel<<endl;;
}
```

IMPORTANT MODELS

DEFINE A CLASS:

1. Define a class student with the following specifications:

Private members of class student:

Admno	integer
Sname	20 character
English	float
Math	float
Science	float
Total	float
Ctotal( )	A function to calculate English + math + science with float return type

Public member functions of class student:
Takedata( ):Function to accept values for admno,sname, English, math, science and invoke ctotal to calculate total.
Showdata( ):Function to display all the data members on the screen.

```
class student
```

```
{    int Admno;
     char Sname[20];
          float English,Math,Science,Total;
          float Ctotal()
          {   Total=English+math+science;
              return Total;
          }
          public:
         void Takedata()
          {  cout<<"\nEnter the admission
                number,name of the student: ";
            cin>>Admno;
            gets(sname);
            cout<<"\nEnter English, Maths,
                          Science Marks: ";
             cin>>English>>Math>>Science;
             Ctotal( );
         }
         void Showdata( )
         {   cout<<"\nThe admission number of
                    the student: "<<Admno;
             cout<<"\nThe name of the student:
                                "<<Sname;
             cout<<"\nEnglish , Maths and
                    Science Marks are…";
             cout<<english<<"\t"<<math<<"\t"
                        <<science<<"\n";
             cout<<"\nTotal marks of the
                        student: "<<Total;
     };
```

REWRITE THE PROGRAM:

1. Rewrite the following program after removing the syntactical errors (if any). Underline each correction. #include [iostream.h]

```
class PAYITNOW
{          int Charge;
           PUBLIC:
            void Raise(){cin>>Charge;}
            void Show{cout<<Charge;}
          };
          void main()
          {      PAYITNOW P;
                P.Raise();
                Show();
          }
```

Answer:

```
          #include <iostream.h>
          class PAYITNOW
          {   int Charge;
          public:
            void Raise(){cin>>Charge;}
            void Show(){cout<<Charge;}
          };
          void main()
          {    PAYITNOW P;
               P.Raise();
               P.Show();
          }
```

Theory Questions:

1. Differentiate between public and private visibility modes. Give suitable example.
2. Differentiate between public and protected visibility modes. Give suitable example.
3. Differentiate between private and protected visibility modes. Give suitable example.
4. Inline Function with examples.