5.CONSTRUCTORS & DESTRUCTORS

MATERIAL

Constructor: A member function with the same name as its class is called Constructor and it is used to initialize the objects of that class type with a legal initial value.

If a class has a constructor, each object of that class will be initialized before any use is made of the object.

Need for Constructors: A variable, an array or a structure in C++ can be initialized at the time of their declaration.

Eg:
```
int a=10;
int a[3]= {5,10,15};
struct student
{
     int rno;
     float m1,m2,m3;
};
student s1={1,55.0,90.5,80.0};
```

But this type of initialization does not work for a class because the class members have their associated access specifiers. They might not be available to the outside world (outside their class). A Constructor is used to initialize the objects of the class being created (automatically called by the compiler).

Difference between a constructor and an ordinary member function:

	Constructor	**Member Function**
Name	Name of the Class	Any Valid Identifier
Purpose	Initialize the object when it is being created	For any general purpose
Call	Implicit	Explicit
Return type	Should not keep	Must be there at least void

Declaration and Definition:

A constructor is a member function of a class with the same name as that of its class name. A constructor is defined like other member functions of a class. It can be defined either inside the class definition or outside the class definition.

Eg:
```
class X
   { int i;
public:
   int j,k;
   X ( )                        //Constructor
    {   i = j = k = 0;
    }
   ------
       //Other members
     ------
    };
```

This simple constructor (X::X ( )) is as an inline member function. Constructors can be written as outline functions also as it is shown below:

```
class X
{    int i ;
  public:
     int j, k ;
     X (  );       //Only constructor declaration.
      ------       //Other members
      ------
 };
X :: X ( )     //Constructor defined outside
{
        i = j = k = 0;
 }
```

Generally constructor will be defined under public section, which can be available to non members also. But it can also be defined under private or protected. A private or protected constructor is not available to the non-member functions. Ie With a private or protected constructor, you cannot create an object of the same class in a non-member function.

There are three types of constructors

a) **Non-parameterized or Default Constructor**
b) **Parameterized Constructor**
c) **Copy Constructors**

a) Default constructor:

A constructor that accepts no parameter is called the default constructor.
With a default constructor, objects are created just the same way as variables of other data types are created.

```
class X
{    int i ;
  public:
     int j, k ;
      ------
     //Members Functions
      ------
 };
```

Eg: X ob1;
    Student s1;

If a class has no explicit constructor defined, the compiler will supply a default constructor. This implicitly declared default constructor is an **inline public** members of its class. Declaring a constructor with arguments hides the default constructor.

There can be a default constructor as well as another constructor with arguments for a class, having multiple constructors is called as constructor overloading.

A constructor can also have default arguments. A constructor with default arguments is equivalent to a default constructor.

Eg: class Rectangle
```
    { float  l,b,a;
      public:
      Rectangle ( float len = 5.0, float bre = 5.0)
          //Constructor with Default arguments
       {  l = len;
          b = bre;
        }
        -----
        -----
    };
 void main( )
 {   Rectangle  first(7.0,9.5);
    Rectangle  second;
//Takes default argument values.  Equivalent to second(5.0,5.0)
    ----
    ----
 }
```
The default constructors are very useful when you want to create objects without having to type the initial objects every time with pre specified initial values or if you want to create array of objects of your class type. You can't create an array of objects unless your class has a default constructor (implicitly or explicitly defined).

b) Parameterized Constructor:

A constructor that take arguments, is called as parameterized constructor.

The parameterized constructor allow us to initialize the various data elements of different objects with different values when they are created. This is achieved by passing different values as arguments to the constructor function when the objects are created.

Eg: class Rectangle
```
       {     float  l,b,a;
       public:
            Rectangle ( float len , float bre )
               //Parameterized Constructor.
            {     l = len;
                  b = bre;
            }
            -----
       };
       void main( )
       {
            Rectangle  first(7.0,9.5);
            ----
            ----
        }
```
With a parameterized constructor, the initial values must be passed at the time of object created. This can be done in two manners:

(i)By calling the constructor implicitly (implicit call)
Eg: Rectangle first(8.5,3.9);

(ii)By calling the construct or explicitly (Explicit call)
Eg: Rectangle first = Rectangle (8.5,3.9);

Temporary Instances:

A temporary instance lives in the memory as long it is being used or referenced in an expression and after this it dies. A temporary instance will not have any name. The explicit call to a constructor also allows you to create a temporary instance or temporary object. The temporary instances are deleted when they are no longer referenced.

Eg: class Sample
```
       {   int i,j;
        public:
          sample (int a, int b)
          {   i=a;
              j=b;
          }
          void print ( )
          {    cout<<i<<j<<"\n";
          }
          ----
          ----
      };
      void test ( )
      {    Sample S1(2,5);
                 //An object S1 created
           S1.print ( );
                //Data values of S1 printed
           Sample (4,9).print ( );
                //Data values of a  temporary
                //sample instance printed
      }
```
The primitive (fundamental) types also have their own constructors. When no values are provided, they use their default constructors but when you provide initial values, the newly created instance is initialized with the provided value.

Eg: int a,b,c;
```
        //Default constructor used
```

```
int i(3), j(4), k(5);     //i,j,k initialized
```

c) Copy Constructor:

A copy constructor is a constructor of the form **classname(classname &).** The compiler will use the copy constructor whenever you initialize an instance using values of another instance of same type.

Eg:

```
 Sample S1;          //Default constructor used
 Sample S2=S1; //Copy constructor used.  Also
Sample S2(S1);
```

In the above code, for the second statement, the compiler will copy the instance S1 to S2 member by member. If you have not defined a copy constructor, the compiler automatically, creates it and it is public.

A copy constructor takes a reference to an object of the same class an argument.

Eg:

```
  class Sample
  {
      int i,j;
    public:
      Sample (int a, int b)          //Constructor
      {
            i = a;
            j = b;
       }
     Sample (Sample &s)  //Copy Constructor
      {
            i=s.i;
            j=s.j;
            cout<<"Copy constructor
                        Working\n";
      }

      void print( )
      {
            cout<<i<<"\t"<<j<<"\n";
      }
      -----
      -----
  };

void main( )
{
Sample S1(4,9);
          //S1 initialized first constructor used
Sample S2(S1);
          //S1 copied to S2.  Copy constructor called.
Sample S3=S1;
          //S1 coped to S3. Copy constructor called again.
      -----
      -----
}
```

Why the argument to a copy constructor is passed by reference:

If we try to pass the argument by value to a copy constructor (ie, for a class X, if we use an X(X) constructor in place of X(X&), the compiler complaints out of memory. The reason is, when an argument is passed by value, a copy of it is constructed. To create a copy of the object, the copy constructor works. But the copy constructor is creating a copy of the object for itself, thus it calls itself. Again the called copy constructor requires another copy so again it is called. In fact it calls itself again until the compiler runs out of memory. So, in the copy constructor, the argument must be passed by reference, so that to make a copy of the passed object, original object is directly available.

Dynamic initialization of objects: The dynamic initialization means that the initial values may be provided during runtime. The benefit of dynamic initialization is that it provides the flexibility of assigning initial values at run time.

Initialization of Const & Reference Members:

If your class contains a constant and a reference as member field, then you need to specify that through **Member-Initialization List.**

A constructor can initialize the constituent data members of its class through a mem-initialization list that appears in the function header of the constructor.

Eg:

```
class Test
{    int a ;
    char b;
 public:
    Test(int i,char j):a(i), b(j);
            //a(i) initializes member a with value i, b(j)….b with j.
    {
      ….
    }
}
```

You can even have a combination of mem-initialization list and initialization within constructor body.

Eg:

```
class Test
{    ……
  public:
    Test(int i, char j):a(i)
     {
        b=j;
     }
     …..
};
```

And if your class contains a **const** and /or a **reference** member, then these members must be initialized through mem-initialization list as these cannot be initialized within constructor body.

Eg:
```
struct Sname
{ char fname[25];
  char lname[25];
} S1;
class Test
{   int a,b;
    const int max;              //const member
    Sname &name;           //reference member
 public:
    Test ( ):max(300),name(S1)
     {   a=0;
         b=10;
     }
     ------
};
```

Mem-initialization lists are especially used in the following four cases:
(i)initialization of const members.
(ii)initialization of reference members.
(iii)Invoking base class constructor.
(iv)Initialization of member objects.

Constructor Overloading:
The constructor of a class may also be overloaded so that even with different number and types of initial values, an object may still be initialized.

Default Arguments Versus Overloading:

Using default arguments gives the appearance of overloading, because the function may be called with an optional number of arguments.

Eg:
Prototype :
```
float amount (float principal, int time=2, float rate=0.08);
 Can be called as
                  Amount(2000.0,4,0.10);
                  Amount(3520.5,3);
                  Amount(5500.0);
```

Special Chracteristics of Constructors:
1. Constructor functions are invoked automatically when the objects are created.
2. If a class has a constructor, each object of that class will be initialized before any use is made of the object.
3. Constructor functions obey the usual access rules. Ie private and protected constructors are available only for member and friend functions, however, public constructors are available for all the functions. Only the functions that have access to the constructor of a class, can create an object of the class.
4. No return type (not even void) can be specified for a constructor.
5. They cannot be inherited, though a derived class can call the base class constructor.
6. A constructor may not be static.
7. Default constructors and copy constructors are generated(by the compiler) where needed. Generated constructors are public.
8. Like other c++ functions, constructors can also have default arguments.
9. It is not possible to take the address of a constructor.
10. An object of a class with a constructor cannot be a
member of a union.
11. Member functions may be called from within a constructor.
12. A constructor can be used explicitly to create new objects of its class type, using the syntax class-name (expression-list)
Eg: Sample obj1=Sample(13,22.42);

DESTRUCTORS

Destructor:

A destructor is used to destroy the objects that have been created by a constructor. A destructor destroys the values of the object being destroyed.

A destructor is also a member function whose name is the same as the class name but is preceded by tilde(~). A destructor takes no arguments, and no return types can be specified for it (not even void). It is automatically called by the compiler when an object is destroyed. A local object, local to a block, is destroyed when the block gets over; a global or static object is destroyed when the program terminates. A destructor cleans up the storage (memory area of the object) that is no longer accessible.

Eg:
```
class Sample
{     int i,j;
   Public:
     Sample(int a, int b)   //Constructor
      {    i=a; j=b;
       }
    ~Sample()
     {   cout<<"Destructor at work\n";
       }
     ------
     ------
};
void main( )
{
  Sample s1(3,4);
//Local object s1 constructed with values 3  & 4 using Sample ( )
 -----
----   /*Automatically s1 is destructed at the end of the block using destructor ~Sample( )*/
}
```

Need for Destructors:

During construction of any object by the constructor, resources may be allocated for use. (for example, a constructor may7 have opened a file and a memory area may be allotted to it). These allocated resources must be de allocated before the object is destroyed.A destructor performs these types of tasks.

Some Characteristics of Destructors:

1. Destructor functions are invoked automatically when the objects are destroyed.
2. If a class has a destructor, each object of that class will be deinitialized before the object goes out of scope.(Local objects at the end of the block defining them and global and static objects at the end of the program).
3. Destructor functions also, obey the usual access rules as other member functions do.
4.No argument can be provided to a destructor, neither does it return any value.
5. They cannot be inherited.
6. A destructor may not be static.
7. It is not possible to take the address of a destructor.
8. Member functions may be called from within a destructor.
9. An object of a class with a destructor cannot be a member of a union.

CONSTRUCTORS AND DESTRUCTORS (PROGRAMS)

1.Program to find area of a circle using class, constructor functions and destructor.

```
#include<iostream.h>
#include<conio.h>
class Circle
{    float  r,a;         //r and a are private
  public:
     Circle()
//Non parameterized or Default Constructor
    { r=0.0;
      a=0.0;
    }
    Circle(float rad)
//Parameterized Constructor
    { r = rad;
      a = 3.1415*r*r;
    }
   Circle(Circle &obj)  //Copy Constructor
    { r = obj.r;
      a = obj.a;
    }
    ~Circle()
    {cout<<"\nThe object is being destroyed....";
    }
    void take()
    {
      cout<<"Enter the value of Radius: ";
      cin>>r;
    }
   void calculate()
    {
      a = 3.1415*r*r;
    }
  void display()
    { cout<<"\nThe Radius of the Circle = "<<r;
      cout<<"\nThe Area of the Circle = "<<a;
    }
};
void main()
{   clrscr();
  Circle c1; /*Default Constructor will be called implicitly.  ie c1.r = 0.0 and c1.a = 0.0  */
  Circle c2(10.3);
           //Parameterized Constructor will be called implicitly
  Circle c3(c2);
           //Copy Constructor will be called implicitly
  c1.take();
  c1.calculate();
  c1.display();
  c2.display();
  c3.display();
  getch();
}
```

2. Program to process student data using class concept, constructors and destructor.

```
#include<iostream.h>
#include<conio.h>
class Student
{   float m1,m2,m3,total,avg;
 public:
   Student()
   { m1=0.0;
     m2=0.0;
     m2=0.0;
     total=0.0;
     avg=0.0;
   }
  Student(float x,float y,float z)
   { m1=x;
     m2=y;
     m3=z;
     total=m1+m2+m3;
     avg=total/3;
   }
 Student(Student &Test)
    { m1=Test.m1;
      m2=Test.m2;
      m3=Test.m3;
      total=Test.total;
      avg=Test.avg;
    }
   ~Student()
    {  cout<<"The Object is being Destroyed....";
    }
    void readProcess()
    { cout<<"\nEnter the 3 Subject marks of a
                                    student: ";
      cin>>m1>>m2>>m3;
      total=m1+m2+m3;
      avg=total/3;
    }
 void display()
    { cout<<"\nTotal Marks = "<<total;
      cout<<"\nAverage Marks = "<<avg;
    }
 };
void main()
 {
    clrscr();
    Student S1;
    Student S2(50.5,90.0,75.5);
    Student S3=S2;
    S1.readProcess();
    S1.display();
    S2.readProcess();
    S2.display();
    S3.display();    getch();
}
```

2010 Delhi:

(b) Answer the questions (i) and (ii) after going through the following class: **2**

```
class TEST
{
        int Regno, Max, Min, Score;
   public:
       TEST() //Function 1
        {
           Regno= 101;
           Max=100;
           Min=40;
           Score=75;
        }
        TEST(int Pregno,int Pscore) //Function 2
        {
            Regno=Pregno;
            Max=100;
            Min=40;
            Score=Pscore;
         }
         ~TEST() //Function 3
         {
            cout<<"TEST Over"<<endl;
         }
void Display() //Function 4
{
 cout<<Regno<<":"<<Max<<":"<<Min<<endl;
      cout<<"[Score]"<<Score<<endl;
}
};
```

(i) As per Object Oriented Programming, which. concept is illustrated by **Function 1 and Function 2** together?
Ans.
Polymorphism (OR) Function Overloading (OR) Constructor Overloading
(ii) What is **Function 3** specifically referred as ? When do you think, **Function 3** will be invoked/called?
Ans.
Destructor, invoked or called when scope of an Object

2010 Outside Delhi:

2. (b) Answer the questions (i) and (ii) after going through the following class: 2

```
class Exam
{
   int Rno,MaxMarks,MinMarks,Marks;
public:
   Exam ( ) //Module 1
   {
    Rno=101;
    MaxMarks=l00;
      MinMarks=40;
      Marks=75;
    }
    Exam (int Prno, int Pmarks) //Module 2
    { Rno=Prno;
      MaxMarks=l00;
      MinMarks=40;
      Marks=Pmarks;
     }
    ~Exam () //Module 3
     {
        cout<<"Exam Over"<<endl;
     }
      void Show () //Module 4
      {
  cout<<Rno<<":"<<MaxMarks<<":"<<MinMarks<<endl
;
      cout<<"[Marks Got]"<<Marks<<endl;
       }
};
```

(i) As per Object Oriented Programming, which concept is illustrated by**Module 1** and **Module 2** together?
Ans.

Polymorphism (OR) Constructor Overloading (OR) Function Overloading

(ii) What is **Module 3** referred as ? When do you think, **Module 3** will be invoked/called?

Ans.

Destructor. It is invoked as soon as the scope of the object gets over.

2009 Delhi:

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

```
#include [iostream.h]
#include [stdio.h]
class Employee
{
int EmpId=901;
char EName [20] ;
public
Employee(){}
void Joining() {cin>>EmpId; gets (EName);}
void List () {cout<<EmpId<<" :
"<<EName<<endl;}
};
void main ()
{
Employee E;
Joining.E();
E.List()
}
```

Ans

```
#include <iostream.h>
#include <stdio.h>
class Employee
{           int EmpId;
            char EName[20];
     public :
            Employee()
            {
                  EmpId=901;
              }
            void Joining( )
            {
                 cin>>EmpId;
                 gets (EName);
            }
              void List ( )
              {
                  cout<<EmpId<<":
"<<EName<<endl;
              }
};
void main ()
{
            Employee E;
            E.Joining ();
            E.List ();
}
```

2.(a) What is copy constructor?Give ane example in C++ to illustrate copy con-structor.2

Ans A copy constructor is an overloaded constructor function in which (an) object(s) of the same class is/are passed as a reference parameter(s). It is used when an object's data value is related to or is initialised using another object's data value of the same class. In the example below the values of data members of object Q are dependent on the values of data members of object P and Data members of object R dependent on Q.

```
//Example of Copy Constructor
class Play
{
            int Count, Number;
     public:
            Play(); //constructor
            Play(Play &);//copy constructor
            void Disp();
            void Change(int,int);
};
Play::Play () //constructor
{
            Count=0;
            Number=0;
}
Play:: Play (Play &P) //copy constructor
{
            Count=P.Count+l0;
            Number=P.Number+20;
}
void Play::Disp()
{
            cout<<Count;
            cout<<Number<<endl;
}
void Play::Change(int C,int N)
{
            Count=C;
            Number=N;
}
void main ()
{
            Play P; //Call for constructor
            P.Disp (); P.Change(90,80) ;
            Play Q(P); //Copy constructor call
            Q.Disp();
            Play R=Q; //Copy constructor ca11
                        [same as P1ay R(Q);]
            R. Disp();
}
```

(b) Answer the questions (i) and (ii) after going through the following class:

```
class WORK 2
{
        int WorkId;char WorkType ;
public:
        -WORK ( ) //Function 1
        {
           cout<<"Un-Allocated"<<endl ;
        }
        void status ( ) //Function 2
        {
      cout<<WorkId<<":
"<<WorkType<<endl ;
          }
        WORK ( ) //Function 3
        {
          WorkId = 10;
         WorkType='T' ;
        }
        WORK(WORK &W) //Function 4
        {
              WorkId=W. WorkId+12;
              WorkType=W. WorkType+l
          }
} ;
```

(i) Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class WORK is called automatically, when the scope of an object gets over? Is it known as Constructor OR Destructor OR Overloaded Function OR Copy Constructor?

Ans Function 1
Destructor.

(ii) WORK W; // Statement 1
WORK Y(W); // Statement 2

Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class WORK will be called on execution of statement written as statement 2 ? What is this function specifically known as
out of Destructor or Copy Constructor or Default Constructor?

Ans Function 4
Copy Constructor.

2009 Outside Delhi:

(b) Answer the questions (i) and (ii) after going through the following class: 2

```
class Job
{int JobId;
        char JobType;
public:
        ~Job ( ) //Function 1
        {
          cout<< "Resigned" <<end1;
        }
        Job ( ) //Function 2
        {  JobId=10 ;
          JobType ='T" ;
        }
        void TellMe( )//Function 3
        {
         cout<<JobId<< ": "
<<JobType<<end1;
        }
        Job (Job &J) //Function 4
        {
          JobId=J.JobId+10;
          JobType=J.JobType+l;
        }
};
```

(i) Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class Job is called automatically, when the scope of an object gets over? Is it known as Constructor OR Destructor OR Overloaded Function OR Copy Constructor?

Ans Function 1.
Destructor.

(ii) Job P ;   //Line 1
    Job Q(P) ;   //Line 2

Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class Job will be called on execution of statement written as Line 2 ? What is this function specifically known as out of Destructor or Copy Constructor or Default Constructor?

Ans Function 4.   Copy Constructor.

2008 DELHI:

2.b) Answer the questions (i) and (ii) after going through the following program: 2

```
#include <iostream.h>
#include<string.h>
class bazaar
{   char Type[20] ;
    char product [20];
    int qty ;
    float price ;
    bazaar()                          //function 1
    {   strcpy (type , "Electronic") ;
        strcpy  (product , "calculator");
        qty=10;
        price=225;
    }
  public  :
  void Disp()       //function 2
 {   cout<< type <<"-"<<product<<":"
          <<qty<< "@" << price << endl ;
 }
};
void main ()
{     Bazaar  B ;       //statement 1
     B. disp() ;       //statement  2
```

}

(i) Will statement 1 initialize all the data members for object B with the values given in the function 1 ? (YES OR NO).
Justify your answer suggesting the correction(s) to be made in the above code.
Ans: No. The reason is the constructor should be defined under the public visibility label.
(ii) What shall be the possible output when the program gets executed ? (Assuming, if required _ the suggested correction(s) are made in the program).
Ans: Possible Output:

```
Electronic–Calculator:10@225
```

2.c) Define a class Garments in c++ with following descriptions. 4

private members :

GCode	of type string
GType	of type string
Gsize	of type intiger
Gfabric	of type istring
Gprice	of type float

A function **Assign()** which calculate and the value of GPrice as follows.
For the value of GFabric "COTTON" ,

GType	**GPrice(RS)**
TROUSER	1300
SHIRT	1100

For GFabric other than "COTTON", the above mentioned
GPrice gets reduced by 10%

public members:
A constructor to assign initial values of GCode,GType and GFabric with the a word "NOT ALLOTED"and Gsize and Gprice with 0.
A function Input ()to the values of the data membersGCode, GType,Gsize and GFabric and invoke the Assign() function.
A function Display () which displays the content of all the data members for a garment.

```
#include<iostream.h>
#include<string.h>
#include<conio.h>
#include<stdio.h>
class Garments
{    char GCode[21],GType[21];
     int Gsize;
     char Gfabric[21];
     float Gprice;
     void Assign( )
     {
if(strcmp(strupr(Gfabric),"COTTON")==0)
  { if(strcmp(strupr(GType),"TROUSER")==0)
                        Gprice=1300;
    if(strcmp(strupr(GType),"SHIRT")==0)
                        Gprice=1100;
  }
 else
  {if(strcmp(strupr(GType),"TROUSER")==0)
                        Gprice=1300*0.90;
  if(strcmp(strupr(GType),"SHIRT")==0)
                        Gprice=1100*0.90;
  }
 }
public:
   Garments( )
   {
     strcpy(GCode,"NOT ALLOTED");
            strcpy(GType,"NOT ALLOTED");
            Gsize=0;
            strcpy(Gfabric,"NOT ALLOTED");
            Gprice=0;
 }
 void Input( )
  {  cout<<"\nEnter the Grament Code: ";
    gets(GCode);
    cout<<"\nEnter the Garment Type: ";
    gets(GType);
    cout<<"\nEnter the Garment Size: ";
    cin>>Gsize;
    cout<<"\nEnter the Garment Fabric: ";
    gets(Gfabric);
    Assign( );
  }
  void display( )
  {  cout<<"\nThe Garment Code: "<<GCode;
    cout<<"\nThe Garment Type: "<<GType;
    cout<<"\nThe Garment Size: "<<Gsize;
    cout<<"\nThe Garment Fabric: "<<Gfabric;
    cout<<"\nThe Garment Price: "<<Gprice;
  }
 };
 void main( )
  {  Garments G;
    G.Input( );
    G.display( );
}
```

2008 OUTSIDE DELHI:

2.b) Answer the questions (i) and (ii) after going through the following program:

```
#include<iostream.h>
#include<string.h>
class Retail
{  char category[20];
   char item[20];
   int  qty;
   float price;
  retail ()                   //function 1
  {   strcpy (category,  "cerial");
      strcpy (Item, "Rice");
      qty =100 ;
      price =25 ;
   }
 public;
   void show()                      //function 2
  {  cout << category <<"-"<< Item << "
             :"<<Qty<<"@"<< price<<endl;
   }
};
void main()
{   Retail R;                 //statement 1
    R. show ();              //statement 2
}
```

(i) will statement 1 initialize all the data members for objects R with the given in the function 1 ? (YES OR NO). Justify your Answer suggesting the corrections(s) to be made in the above code.

Ans: No. The reason is the constructor should be defined under the public visibility label.

(ii) What shall be the possible out put when the program gets executed ? (Assuming, if required the suggested correction(s) are made in the program)

Ans: Possible Output:

```
            cerial–Rice:100@25
```

2.c) Define a class clothing in c++ with the following descriptions :

private members :

code	of type string
type	of type string
size	of type intiger
material	of type string
price	of type float

A function **calc_price( )**which calculates and assigns the value of GPrice as follows ;

For the value of material as "COTTON" :

Type	**price (Rs)**
TROUSER	1500.
SHIRT	1200.

for material other than "COTTON", the above mentioned GPprice price gets reduced by 25%

public members :

* A constructor to assign initial values of code ,type and material with the word "NOT ASSIGNED "and size and price with 0.
* A function enter() to input the values of the data members code, type, size and material and invoke the caclPrice () function.
* A function show which displays the content of all the data members for a clothing.

```
#include<iostream.h>
#include<string.h>
#include<conio.h>
#include<stdio.h>
class clothing
{    char Code[21],Type[21];
     int size;
     char material[21];
     float price;
     void calc_price( )
     {
if(strcmp(strupr(material),"COTTON")==0)
 {   if(strcmp(strupr(Type),"TROUSER")==0)
                          price=1500;
    if(strcmp(strupr(Type),"SHIRT")==0)
                 price=1200;
 }
else
  {      if(strcmp(strupr(Type),"TROUSER")==0)
                          price=1500*0.75;
        if(strcmp(strupr(Type),"SHIRT")==0)
                          price=1200*0.75;
   }
 }
public:
  clothing( )
  {   strcpy(Code,"NOT ALLOTED");
     strcpy(Type,"NOT ALLOTED");
     size=0;
     strcpy(material,"NOT ALLOTED");
     price=0;
  }
  void enter( )
  {
     cout<<"\nEnter the Cloth Code: ";
     gets(Code);
     cout<<"\nEnter the Cloth Type: ";
     gets(Type);
     cout<<"\nEnter the Cloth Size: ";
     cin>>size;
    cout<<"\nEnter the cloth material: ";
     gets(material);
     calc_price( );
  }
  void show( )
  {
     cout<<"\nThe Cloth Code: "<<Code;
     cout<<"\nThe Cloth Type: "<<Type;
     cout<<"\nThe Cloth Size: "<<size;
     cout<<"\nThe Cloth Material: "
                              <<material;
     cout<<"\nThe Cloth Price: "<<price;
            }
};
void main( )
{
            clothing C;
            C.enter( );
            C.show( );
}
```

2007 DELHI:

2.a) Differentiate between Constructor and Destructor function in context of Classes and Objects Using C++?   2

Ans: Constructor: A constructor is used to intitialize the objects of that class type with a legal initial value.If a class has a constructor, each object of that class will be initialized before any use is made of the object.

(A member function with the same name as its class is called Constructor and it is used to initialize the objects of that class type with a legal initial value.)

Destructor: A destructor is used to destroy the objects that have been created by a constructor. A destructor destroys the values of the object being destroyed.

Constructor	Destructor
Purpose: Is used to intitialize the objects of that class type with a legal initial value	**Purpose:** Is used to destroy the objects that have been created by a constructor
Name: The name of the class	**Name:**The name of the class preceded by a ~.
Calling: It will be called automatically at the time of creation of the object. Ie Implicite calling	**Calling:** It will be called automatically at the time of destruction of an object. Ie Implicite calling
Return Type: No return type not even void	**Return Type:** No return type not even void

2.b) Answer the question (i)and (ii)after going through the following class: 2

```
 class Maths
{      char Chapter[20]
       int Marks;
public:
       Maths()                    //Member Function 1
       {   strcpy (Chapter, "Geometry");
           Marks=10;
           cout <<"Chapter Initialised ";
       }
       -Maths()                   //Member Functions 2
       {   cout<<"Chapter Over";
       }
 };
```

(i) Name the specific features of class shown by member Function 1 and Member Function 2 in the above example.

Ans: Member function 1 is a (non-parameterized or default) constructor (, which will be executed automatically at the time of creation of an object of class Maths).

Member function 2 is a destructor (,which will be executed automatically at the time of destruction of an object of class Maths).

(ii) How would Member Function 1 and Member Function 2 get executed ?

Ans: They will be executed automatically.

Member function 1 will be executed at the time of creation of an object of class Maths. Member function 2 will be executed at the time of destruction of an object of class Maths.

2.c) Define a class Tour in C++ with the description given below. 4

Private Members:

TCode	of type string
No of Adults	of type integer
No of Kids	of type integer
Kilometers	of type integer
TotalFare	of type float

Public Members:

- A constructor to assign initial values as follows:
  TCode with the word "NULL"
  No of Adults as 0
  No of Kids as 0
  Kilometers as 0
  TotalFare as 0
- A function AssignFare() which calculates and assigns the value of the data member Totalfare as follows
  For **each** Adult

Fare (Rs)	*For Kilometers*
500	>=1000
300	<1000 & >=500
200	<500

  For **each** Kid the above Fare will be 50% of the Fare mentioned in the above table
  For Example:
  If Kilometers is 850, Noofadults =2 and NoofKids =3 Then TotalFare should be calculated as
  Numof Adults *300+ NoofKids *150
  i.e., 2*300+ 3 *150 =1050
- A function EnterTour() to input the values of the data members TCode, NoofAdults, NoofKids and Kilometers ; and invoke the AssignFare() function.
- A function ShowTour() which displays the content of all the data members for a Tour.

Ans:

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
#include<iostream.h>
class Tour
{  char TCode[21];
  int NoofAdults,NoofKids,Kilometres;
  float TotalFare;
 public:
  Tour( )
  { strcpy(TCode,"NULL");
NoofAdults=NoofKids=Kilometres=TotalFare=0;
  }
  void AssignFare( )
 { if(Kilometres>=1000)
TotalFare=NoofAdults*500+NoofKids*250;
   else if(Kilometres>=500)
TotalFare=NoofAdults*300+NoofKids*150;
   else
TotalFare=NoofAdults*200+NoofKids*100;
 }
 void EnterTour( )
 {  cout<<"\nEnter the Tour Code: ";
   gets(TCode);
   cout<<"\nEnter the Number of Adults: ";
   cin>>NoofAdults;
   cout<<"\nEnter the Number of Kids: ";
   cin>>NoofKids;
  cout<<"\nEnter the Number of Kilometres: ";
   cin>>Kilometres;
   AssignFare( );
 }
 void ShowTour( )
 {    cout<<"\nThe Tour Code: "<<TCode;
   cout<<"\nThe Number of Adults:"
                         <<NoofAdults;
   cout<<"\nThe Number of Kids:
                         "<<NoofKids;
   cout<<"\nThe Number of Kilometres: "
                        <<Kilometres;
   cout<<"\n\nThe Total Fare: "<<TotalFare;
 }
};
void main( )
{  clrscr();
   Tour T;
   T.EnterTour( );
   T.ShowTour( );
   getch();
}
```

2007 OUTSIDE DELHI:

2.b) Answer the questions (i) and (ii) after going through the following class : 2

```
   class Science
   {       char Topic[20] ;
           int Weightage ;
   public :
           Science ()                  //Function 1
           {          strcpy (Topic, "Optics") ;
                      Weightage =30
                      cout<<"Topic Activated";
           }
           ~Science()                  //Function 2
           {          cout<<"Topic  Deactivated";   }
};
```

(i)Name the specific features of class shown by Function 1 and Function 2 in the above example.

Ans: Member function 1 is a (non-parameterized or default) constructor (, which will be executed automatically at the time of creation of an object of class Science).

Member function 2 is a destructor (,which will be executed automatically at the time of destruction of an object of class Science).

(ii) How would Function 1 and Function 2 get executed ?

Ans: They will be executed automatically. Member function 1 will be executed at the time of creation of an object of class Science. Member function 2 will be executed at the time of destruction of an object of class Science.

2.c) Define a class Travel in C++ with the description given below : 4

Private Members:

T_Code	of type string
No_ of_ Adults	of type integer
No _of _Children	of type integer
Distance	of type integer
TotalFare	of type float

Public Members:

- A constructor to assign initial values as follows:
  TCode with the word "NULL"
  No _of_ Adults as 0
  No_ of_Children as 0
  Distance as 0
  TotalFare as 0
- A function AssignFare() which calculates and assigns the value of the data member Totalfare as follows
  For **each** Adult

Fare (Rs)	*For Kilometers*
500	>=1000
300	<1000 & >=500
200	<500

For **each** Child the above Fare will be 50% of the Fare mentioned in the above table
For Example:
If Distance is 750, No_of_adults =3 and No_of_Children =2
Then TotalFare should be calculated as
Num_of _Adults *300+ No_of_Children *150
i.e., 3*300+ 2 *150 =1200

- A function EnterTour() to input the values of the data members T_Code, No_of_Adults, No_of_Children and Distance ; and invoke the AssignFare() function.
- A function ShowTravel() which displays the content of all the data members for a Travel.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
#include<iostream.h>
class Travel
{  char T_Code[21];
   int No_of_Adults,No_of_Children,Distance;
   float TotalFare;
 public:
    Travel( )
     {  strcpy(T_Code,"NULL");
   No_of_Adults=No_of_Children=Distance=
                 TotalFare=0;
     }
     void AssignFare( )
     {
 if(Distance>=1000)
      TotalFare=No_of_Adults*500+No_of_Children*250;
 else if(Distance>=500)
      TotalFare=No_of_Adults*300+No_of_Children*150;
 else
      TotalFare=No_of_Adults*200+No_of_Children*100;
     }
     void EnterTravel( )
    {   cout<<"\nEnter the Travel Code: ";
        gets(T_Code);
        cout<<"\nEnter the Number of Adults: ";
        cin>>No_of_Adults;
     cout<<"\nEnter the Number of Children: ";
       cin>>No_of_Children;
     cout<<"\nEnter the Distance in Kilometres: ";
      cin>>Distance;
      AssignFare( );
     }
   void ShowTravel( )
   { cout<<"\nThe Travel Code: " <<T_Code;
cout<<"\nThe Number of Adults: "
<<No_of_Adults;
 cout<<"\nThe Number of Children:
"<<No_of_Children;
 cout<<"\nThe Distance in Kilometres: "<<Distance;
 cout<<"\n\nThe Total Fare: "<<TotalFare;
    }
 };
 void main( )
 {
   clrscr();
   Travel T;
   T.EnterTravel( );
   T.ShowTravel( );
   getch();
 }
```

2006 DELHI:

2.b) Answer the following questions (i) and (ii) after going through the following class. 2

```
 class Interview
 {    int Month;
   public:
      interview(int y)   {Month=y;}//constructor 1
     interview(Interview&t);          //constructor 2
  };
```

(i) create an object, such that it invokes Constructor 1.

Ans: Interview A(10); //invoking constructor 1 by passing a number.

(ii) write complete definition for Constructer 2.

Ans:
```
Interview(Interview  &t)
        //This is a copy constructor.
     {         Month=t.Month;
     }
```

2006 OUTSIDE DELHI:

1.f) What is a default constructor? How does it differ from destructor? 2

a) Default constructor: A constructor that accepts no parameter is called the default constructor.

With a default constructor, objects are created just the same way as variables of other data types are created.

```
     class X
   {      int i ;
     public:
        int j, k ;
         ------                    //Members
Functions
         ------
    };
```

Eg: X ob1;
 Student s1;

If a class has no explicit constructor defined, the compiler will supply a default constructor. This implicitly declared default constructor is an **inline public** members of its class. Declaring a constructor with arguments hides the default constructor.

There can be a default constructor as well as another constructor with arguments for a class, having multiple constructors is called as constructor overloading.

2.b) Answer the following questions (i) and (ii) after going through the following class. 2

```
     class Exam
     {      int Year;
        public:
          Exam(int y)     //Constructor 1
          {         Year=y;
          }
          Exam(Exam &t);
                //Constructor 2
     };
```

(i) Create an object, such that it invokes Constructor 1

Ans: Exam E((2008);

(ii) Write complete definition for constructor 2.

Ans: Exam(Exam &t)

```
            //Copy Constructor.
           {      Year=t.Year;
           }
```

2005 DELHI:

2.b) Answer the following questions (i) and (ii) after going through the following class.

```
          class Test
          {     char Paper[20];
                int Marks
          public:
                Test()      //Function 1
                {     strcpy(Paper,"Computer");
                      Marks=0;
                }

            //Function 2
                Test(char P[])
                {     strcpy(Paper,P);
                      Marks=0;
                }

           //Function 3
                Test(int M)
                {     strcpy(Paper,"Computer");
                      Marks=M;
                 }
                 Test(char P[],int M)
                  //Function 4
                {     strcpy(Paper,P);
                      Marks=M;
                }
          };
```

(i)Which feature Object Oriented programming is demonstrated using Function 1, Function 2, Function 3 and Function 4 in the above class text?

Ans: Function overloading (here it is constructor overloading).

(ii)Write statements in C++ that would execute Function 2 and Function 4 of class Text.

Ans: **(let** char name[20];

```
              int X=60;
       strcpy(name,"COMPUTERSCIENCE");
        are declared in the program)
  (i)     Test  A(name);
          //Will execute Funciton 2
  (ii)    Test  B(name,X);
         //Will execute Function 4
```

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

Private Members:

Plancode	of type long
Place	of type character array(string)
Number_of_travellers	of type integer
Number_of_buses	of type integer

Public Members:

*A constructer to assign initial values of PlanCode as 1001, Place as "agra",Number_of_travellers as 5,Number_of_buses as 1

* A function NewPlan() which allows user to enter PlanCode, Place and Number_of travelers. Also, assign the value of Number_of_buses as per the following conditions:

Number_of_travellers	**Number_of_buses**
	1
less than 20	1
Equal to or more than 20 and less than 40	2
Equal to 40 or more than 40	3

* A function ShowPlan() to display the3 content of all the data members on the screen.

Ans:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
class TravelPlan
{   long PlanCode;
  char Place[21];
  int Number_of_travellers,Number_of_buses;
 public:
  TravelPlan( )
  {  PlanCode=1001;
    strcpy(Place,"Agra");
    Number_of_travellers=5;
    Number_of_buses=1;
  }
  void NewPlan( )
  {  cout<<"\nEnter the Plan Code: ";
    cin>>PlanCode;
    cout<<"\nEnter the Place to Travel: ";
    gets(Place);
   cout<<"\nEnter the Number of Travellers: ";
    cin>>Number_of_travellers;
    if(Number_of_travellers>=40)
            Number_of_buses=3;
    else if(Number_of_travellers>=20)
           Number_of_buses=2;
    else
           Number_of_buses=1;
  }
  void ShowPlan( )
  {  cout<<"\nThe Plan Code: "<<PlanCode;
    cout<<"\nThe Place of Travel: "<<Place;
    cout<<"\nNumber of Travellers: "
                  <<Number_of_travellers;
    cout<<"\nNumber of Buses: "
                 <<Number_of_buses;
  }
 };
 void main( )
 {   clrscr( );
    TravelPlan T;
    T.NewPlan( );
    T.ShowPlan( );
    getch();
```

}

2005 OUTSIDE DELHI:

1.a) Differentiate between a default constructer and copy constructer, giving suitable examples of each.

Ans: A default constructor also called as non-parameterized constructor will take no argument and initialize the object with the predefined values in that constructor,

Where as a copy constructor will take an already created object of that class and stores that object values into the newly created object of that class. A copy constructor takes a reference to an object of the same class as an argument.

2.b) Answer the following questions (i)and (ii) after going through the following class.

```
class Exam
{           int Marks;
            char Subject[20];
public:
     Exam()     //Function 1
     {     strcpy(Subject,"Computer");
           Marks=0;
      }
     Exam(char S[])  //Function 2
     {     strcpy(Subject,S);
          Marks=0;               }
     Exam(int M)           //Function 3
     {    strcpy(Subject,"Computer");
         Marks=M;
      }
      Exam(char S[],int M) //Function4
      {    Strcpy(Subject,P);
          Marks=M;
      }
};
```

(i)Write statements in C++ that would execute Function 3 and Function 4 of class Exam.

```
     (let    char name[20];
                int X=60;
   strcpy(name,"COMPUTERSCIENCE");
          are declared in the program)
    (i)      Exam  A(X);
          //Will execute Funciton 3
    (ii)     Exam  B(name,X);
          //Will execute Function 4
```

(ii)Which feature Object Oriented Programming is demonstrated using Function 1, Function 2, Function 3 and Function 4 in the above class text?

Ans: Function overloading (here it is constructor overloading).

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

Private Members:

Travelcode	of type long
Place	of type character array(string)
Number_of_travellers	of type integer
Number_of_buses	of type integer

Public Members:

* A constructer to assign initial values of TravelCode as 201,
Place as "Nainital", Number_of_travellers as 10, Number_of_buses as 1

* A function NewTravel() which allows user to enter TravelCode, Place and Number_of travelers. Also, assign the value of Number_of_buses as per the following conditions:

Number_of_travellers	**Number_of_buses**
	1
less than 20	1
Equal to or more than 20 and less than 40	2
Equal to 40 or more than 40	3

* A function ShowTravel() to display the content of all the data members on the screen.

Ans:

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
class Travel
{  long TravelCode;
   char Place[21];
   int No_of_travellers,No_of_buses;
 public:
   Travel( )
   {  TravelCode=201;
      strcpy(Place,"Nainital");
      No_of_travellers=5;
      No_of_buses=1;
   }
   void NewTravel( )
   {  cout<<"\nEnter the Travel Code: ";
      cin>>TravelCode;
      cout<<"\nEnter the Place to Travel: ";
      gets(Place);
    cout<<"\nEnter the Number of Travellers: ";
      cin>>No_of_travellers;
      if(No_of_travellers>=40)
            No_of_buses=3;
      else if(No_of_travellers>=20)
            No_of_buses=2;
      else
           No_of_buses=1;
   }
   void ShowTravel( )
   {  cout<<"\nThe Plan Code: "<<TravelCode;
      cout<<"\nThe Place of Travel: "<<Place;
      cout<<"\nNumber of Travellers: "
               <<No_of_travellers;
      cout<<"\nNumber of Buses: "
                <<No_of_buses;
   }
};
void main( )
{   clrscr( );
    Travel T;
    T.NewTravel( );
    T.ShowTravel( );
    getch();
}
```

2004 DELHI:

2.a) Given the following C++ code, answer the questions (i)and(ii)

```
      class TestMeOut
      {   public:
        ~TestMeOut( )   //Function 1
         {
 cout<<"Leaving the examination hall"<<endl;
         }
         TestMeOut( )  //Function 2
         {
cout<<"Appearing for examination"<<endl;
         }
         void MyWork( )
         {
 cout<<"Attempting Questions"<<endl;
         }
     };
```

(i) In Object Oriented programming, what is Function 1 referred as and when does it get invoked/called?

Ans: Function 1 is called as Destructor, It will automatically executed at the time of destruction of the object of class TestMeOut.

(ii) In Object Oriented Programming, what is Function 2 referred as and when does it get invoked/called?

Ans: Function 2 is called as constructor (Non-parameterized or default constructor) , it will automatically executed at the time of creation of the object of class TestMeOut.

2003 DELHI:

2.b) Define a class **Play** in C++ with the following specifications:

Private members of class **Play**

*Play code	integer
*Playtime	25 character
*Duration	float
*Noofscenes	integer

Public member function of class Play

*A constructer function to initialize Duration as 45 and Noofscenes as

*Newplay() function to values for Playcode and Playtitle.

*Moreinfor() to assign the values of assign the values of Duration and Noofscenes with the of corresponding values passed as parameters to this function.

*Shoplay() function to display all the dataq members on the screen.

Ans:
```
#include<iostream.h>
        #include<conio.h>
        #include<string.h>
        #include<stdio.h>
            class Play
            {   int Playcode;
               char Playtitle[25];
               float Duration;
               int Noofscenes;
             public:
               Play( )
               {  Duration=45;
                  Noofscenes=5;
               }
               void Newplay( )
               { cout<<"\nEnter the Play Code: ";
                  cin>>Playcode;
                  cout<<"\nEnter the Play Title: ";
                  gets(Playtitle);
               }
               void Moreinfor(float D,int N)
               {  Duration = D;
                  Noofscenes = N;
               }
               void Showplay( )
               {  cout<<"\nThe Play Code  : "
                              <<Playcode;
                  cout<<"\nThe Play Title : "
                              <<Playtitle;
                  cout<<"\nThe Duration   :"
                              <<Duration;
                  cout<<"\nThe No of
                          Scenes:"<<Noofscenes;
               }
            };
            void main( )
            {  clrscr( );
              Play P;
              P.Newplay( );
              float Dur;
              int NS;
              cout<<"\nEnter the Duration and
                          Number of Scenes: ";
              cin>>Dur>>NS;
              P.Moreinfor(Dur,NS);
              P.Showplay( );
              getch( );
}
```

2002 DELHI:

2.c) Write the **output** of the following program.4

Ans:
```
#include<iostream.h>
            class Counter
            {  private:
                 unsigned int count;
             public:
                 Counter()
```

Output:
```
C1=0
C2=0
C1=1
C2=2
```

```
            {     count=0;
            }
            void inc_Count()
            {      count++;
            }
            int get_Count()
            {      return count;
            }
       };
       void main()
       {     Counter C1,C2;
            cout<<"\nC1="<<C1.get_Count();
            cout<<"\nC2="<<C2.get_Count();
            C1.inc_Count();
            C2.inc_Count();
            C2.inc_Count();
            cout<<"\nC1="<<C1.get_Count();
            cout<<"\nC2="<<C2.get_Count();
}
```

2000 DELHI:

2.a) Why is destructor function required in classes? Illustrate with the function with an example.

Ans: A destructor is a function which de-allocates/frees the memory which was reserved by the constructor.

```
Eg:
class Sample
{
     Int i,j;
   Public:
     Sample(int a, int b)           //Constructor
     {    i=a; j=b;   }
    ~Sample()
     {     cout<<"Destructor at work\n";    }
     ------
};
void main( )
{
  Sample s1(3,4);    //Local object s1
constructed with values 3
                              // and 4 using Sample ( )
 -----
 ------
 ----//Automatically s1 is destructed at the end
of the block
     //using destructor ~Sample( )
}
```

Here in the above example the destructor ~Sample( ) will be automatically executed at the time of destruction of an object, and which is used to de-allocate the memory, before doing it whatever written in the destructor will be executed.

Ie in the above example whenever an object of the class is being destroyed, "Destructor at work" will be displayed.

1998 DELHI:

2.a) What is a copy constructor? What do you understand by constructer overloading?

Ans: copy constructor is a constructor of the form **classname(classname &).** The compiler will use the copy constructor whenever you initialize an instance using values of another instance of same type.

```
Eg: Sample S1;          //Default constructor used
    Sample S2 = S1;//Copy constructor used.  Also
                        //Sample S2(S1);
```

In the above code, for the second statement, the compiler will copy the instance S1 to S2 member by member. If you have not defined a copy constructor, the compiler automatically, creates it and it is public.

A copy constructor takes a reference to an object of the same class an argument.

Constructor Overloading:

With same constructor name, having several definitions that are differentiable by the number or types of their arguments(ie Parameterized, non-parameterized and copy constructors) is known as an overloaded constructor and this process is known as constructor overloading.

Constructor overloading implements polymorphism.

An Example using Constructor Overloading:

1.Program to find area of a circle using class, constructor functions and destructor.

```
#include<iostream.h>
#include<conio.h>
class Circle
{   float  r,a;        //r and a are private
  public:
    Circle()         //Non parameterized or Default
Constructor
    {  r=0.0;      a=0.0;
   }
    Circle(float rad)     //Parameterized Constructor
    {  r = rad;
      a = 3.1415*r*r;
    }
    Circle(Circle &obj)  //Copy Constructor
    {  r = obj.r;
      a = obj.a;
    }
    ~Circle()
    {      cout<<"\nThe object is being
                destroyed....";
     }
    void take()
    {  cout<<"Enter the value of Radius: ";
       cin>>r;
    }
    void calculate()
    {    a = 3.1415*r*r;
     }
    void display()
    {  cout<<"\nThe Radius of the Circle = "<<r;
      cout<<"\nThe Area of the Circle = "<<a;
    }
};
void main()
{  clrscr();
   Circle c1; /*Default Constructor will be called implicitely.
                ie c1.r = 0.0 and c1.a = 0.0  */
   Circle c2(10.3);  //Parameterized Constructor will be
                     //called implicitely
   Circle c3(c2);
                 //Copy Constructor will be called implicitely
   c1.take();
   c1.calculate();
   c1.display();
   c2.display();
   c3.display();
getch();}
```

Model Paper 1 for 2008-09 Batch:

2.c) Answer the questions (i) and (ii) after going through the following class: 2

```
class Seminar
{
   int Time;
   public:
  Seminar()        //Function 1
  {
  Time=30;cout<<"Seminar starts now"<<end1;
  }
 void Lecture()  //Function 2
 {
 cout<<"Lectures in the seminar on"<<end1;
}
Seminar(int Duration)    //Function 3
{
    Time=Duration;cout<<"Seminar starts
                                   now"<<end1;
}
~Seminar()              //Function 4
 {
     cout<<"Vote of thanks"<<end1;
 }
};
```

i)In Object Oriented Programming, what is Function 4 referred as and when does it get invoked/called?
Answer:

Destructor, it is invoked as soon as the scope of the object gets over.

ii) In Object Oriented Programming, which concept is illustrated by Function 1 and Function 3 together? Write an example illustrating the calls for these functions.
Answer:

Constructor Overloading (Polymorphism)
Seminar S1,S2(90);

Model Paper 2 for 2008-09 Batch:

2.c) Answer the questions (i) and (ii) after going through the following program: 2

```
class Match
{
      int Time;
public:
     Match()                  //Function 1
     {
       Time=0;
       cout<<"Match commences"<<end1;
     }
     void Details()          //Function 2
    {
              cout<<"Inter Section Basketball
Match"<<end1;
     }
     Match(int Duration) //Function 3
     {
       Time=Duration;
   cout<<"Another Match begins now"<<end1;
      }
      Match(Match &M)  //Function 4
      {
        Time=M.Duration;
        cout<<"Like Previous Match "<<end1;
       }
};
```

i)Which category of constructor - Function 4 belongs to and what is the purpose of using it?
Answer:

Copy Constructor, it is invoked when an object is created and initialised with values of an already existing object.

*ii)*Write statements that would call the member Functions 1 and 3
Answer:

```
Match M1;        //for Function 1
Match M2(90);  //for Function 3
```

Sample Paper 1 for 2009-10 Batch:

2. b) Answer the questions (i) and (ii) after going through the following class: 2

```
class Seminar
{ int Time;
  public:
  Seminar()                    //Function 1
  {
   Time=30;
   cout<<"Seminar starts now"<<end1;
   }
   void Lecture() //Function 2
   {
    cout<<"Lectures in the seminar on"<<end1;
   }
   Seminar(int Duration) //Function 3
   {   Time=Duration;
       cout<<"Seminar starts now"<<end1;
   }
   ~Seminar( )//Function 4
  {    cout<<"Vote of thanks"<<end1;
   }
};
```

i) In Object Oriented Programming, what is Function 4 referred as and when does it get invoked/called?
A) Destructor, it is invoked as soon as the scope of the object gets over.
ii) In Object Oriented Programming, which concept is illustrated by Function 1 and Function 3 together? Write an example illustrating the calls for these functions.
A) Constructor Overloading (or Function Overloading or Polymorphism)

```
Seminar S1;                   //Function 1
Seminar S2(90);               //Function 3
```

Sample Paper 2 for 2009-10 Batch:

2.b) Answer the questions (i) and (ii) after going through the following program: 2

```
class Match
{
        int Time;
   public:
        Match()  //Function 1
        {
        Time=0;
        cout<<"Match commences"<<end1;
        }
        void Details() //Function 2
        {
        cout<<"Inter Section Basketball
                   Match"<<end1;
        }
        Match(int Duration) //Function 3
        {
        Time=Duration;
        cout<<"Another Match begins
                        now"<<end1;
        }
        Match(Match &M) //Function 4
        {
        Time=M.Duration;
       cout<<"Like Previous Match "<<end1;
       }
};
```

i) Which category of constructor - Function 4 belongs to and what is the purpose of using it?
A) Copy constructor, It will help to copy the data from one object to another
ii) Write statements that would call the member Functions 1 and 3

```
A) Match M;                    //Function 1
   Match N(10);                //Function 3
```

IMPORTANT MODELS

DEFINE A CLASS:

1. Define a class Garments in c++ with following descriptions
private members :

GCode	of type string
GType	of type string
Gsize	of type intiger
Gfabric	of type istring
Gprice	of type float

A function **Assign()** which calculate and the value of GPrice as follows.
For the value of GFabric "COTTON" ,

GType	**GPrice(RS)**
TROUSER	1300
SHIRT	1100

For GFabric other than "COTTON", the above mentioned
GPrice gets reduced by 10%

public members:
A constructor to assign initial values of GCode,GType and GFabric with the a word "NOT ALLOTED"and Gsize and Gprice with 0.
A function Input ()to the values of the data membersGCode, GType,Gsize and GFabric and invoke the Assign() function.
A function Display () which displays the content of all the data members for a garment.

```
#include<iostream.h>
#include<string.h>
#include<conio.h>
#include<stdio.h>
class Garments
{   char GCode[21],GType[21];
    int Gsize;
    char Gfabric[21];
    float Gprice;
    void Assign( )
    {
if(strcmp(strupr(Gfabric),"COTTON")==0)
  { if(strcmp(strupr(GType),"TROUSER")==0)
                        Gprice=1300;
   if(strcmp(strupr(GType),"SHIRT")==0)
                        Gprice=1100;
  }
 else
  {if(strcmp(strupr(GType),"TROUSER")==0)
                        Gprice=1300*0.90;
  if(strcmp(strupr(GType),"SHIRT")==0)
                        Gprice=1100*0.90;
  }
 }
public:
  Garments( )
  {           strcpy(GCode,"NOT ALLOTED");
              strcpy(GType,"NOT ALLOTED");
              Gsize=0;
              strcpy(Gfabric,"NOT ALLOTED");
              Gprice=0;
 }
 void Input( )
  {  cout<<"\nEnter the Grament Code: ";
    gets(GCode);
    cout<<"\nEnter the Garment Type: ";
    gets(GType);
    cout<<"\nEnter the Garment Size: ";
    cin>>Gsize;
    cout<<"\nEnter the Garment Fabric: ";
    gets(Gfabric);
    Assign( );
  }
  void display( )
  {  cout<<"\nThe Garment Code: "<<GCode;
    cout<<"\nThe Garment Type: "<<GType;
    cout<<"\nThe Garment Size: "<<Gsize;
    cout<<"\nThe Garment Fabric: "<<Gfabric;
    cout<<"\nThe Garment Price: "<<Gprice;
  }
 };
 void main( )
  {  Garments G;
   G.Input( );
   G.display( );
  }
```

2. Define a class **Play** in C++ with the following specifications:

Private members of class **Play**

*Play code	integer
*Playtime	25 character
*Duration	float
*Noofscenes	integer

Public member function of class Play

*A constructer function to initialize Duration as 45 and Noofscenes as

*Newplay() function to values for Playcode and Playtitle.

*Moreinfor() to assign the values of assign the values of Duration and Noofscenes with the of corresponding values passed as parameters to this function.

*Shoplay() function to display all the dataq members on the screen.

Ans:
```
#include<iostream.h>
        #include<conio.h>
        #include<string.h>
        #include<stdio.h>
           class Play
           {    int Playcode;
              char Playtitle[25];
              float Duration;
              int Noofscenes;
            public:
              Play( )
              {  Duration=45;
                 Noofscenes=5;
              }
              void Newplay( )
              { cout<<"\nEnter the Play Code: ";
                 cin>>Playcode;
                 cout<<"\nEnter the Play Title: ";
                 gets(Playtitle);
              }
              void Moreinfor(float D,int N)
              {  Duration = D;
                 Noofscenes = N;
              }
              void Showplay( )
              {  cout<<"\nThe Play Code  : "
           <<Playcode;
                 cout<<"\nThe Play Title : "
           <<Playtitle;
                 cout<<"\nThe Duration   :"
           <<Duration;
                 cout<<"\nThe No of
           Scenes:"<<Noofscenes;
              }
           };
           void main( )
           {  clrscr( );
              Play P;
              P.Newplay( );
              float Dur;
              int NS;
              cout<<"\nEnter the Duration and
                              Number of Scenes:
           ";
              cin>>Dur>>NS;
              P.Moreinfor(Dur,NS);
              P.Showplay( );
              getch( );
}
```

<u>CONSTRUCTOR OVERLOADING:</u>

1. Answer the questions (i) and (ii) after going through the following class:

```
class TEST
{       int Regno, Max, Min, Score;
  public:
     TEST() //Function 1
     {  Regno= 101;
        Max=100;
        Min=40;
        Score=75;
      }
      TEST(int Pregno,int Pscore) //Function 2
      {  Regno=Pregno;
         Max=100;
         Min=40;
         Score=Pscore;
       }
      ~TEST() //Function 3
      {  cout<<"TEST Over"<<endl;
       }
void Display() //Function 4
{     cout<<Regno<<":"<<Max<<":"<<Min<<endl;
      cout<<"[Score]"<<Score<<endl;
}
};
```

(i) As per Object Oriented Programming, which. concept is illustrated by **Function 1 and Function 2** together?

Ans. Polymorphism (OR) Function Overloading
(OR) Constructor Overloading

(ii) What is **Function 3** specifically referred as ? When do you think, **Function 3** will be invoked/called?

Ans. Destructor, invoked or called when scope of an Object

2. Answer the questions (i) and (ii) after going through the following class:

```
class WORK 2
{           int WorkId;char WorkType ;
public:
          -WORK ( ) //Function 1
          {  cout<<"Un-Allocated"<<endl ;
          }
          void status ( ) //Function 2
          {
        cout<<WorkId<<":
"<<WorkType<<endl ;
            }
          WORK ( ) //Function 3
          {   WorkId = 10;
              WorkType='T' ;
          }
          WORK(WORK &W) //Function 4
          {      WorkId=W. WorkId+12;
                 WorkType=W. WorkType+l
            }
} ;
```

(i) Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class WORK is called automatically, when the scope of an object gets over? Is it known as Constructor OR Destructor OR Overloaded Function OR Copy Constructor?

Ans Function 1
Destructor.

(ii) WORK W; // Statement 1
WORK Y(W); // Statement 2

Which member function out of Function 1, Function 2, Function 3 and Function 4 shown in the above definition of class WORK will be called on execution of statement written as statement 2 ? What is this function specifically known as out of Destructor or Copy Constructor or Default Constructor?

Ans Function 4 Copy Constructor.

3. Answer the questions (i) and (ii) after going through the following program:

```
#include <iostream.h>
#include<string.h>
class bazaar
{   char Type[20] ;
    char product [20];
    int qty ;
    float price ;
    bazaar()                          //function 1
    {   strcpy (type ,  "Electronic") ;
        strcpy  (product ,  "calculator");
        qty=10;
        price=225;
    }
  public  :
  void Disp()        //function 2
  {   cout<< type <<"-"<<product<<":"
             <<qty<< "@" << price << endl ;
  }
};
void main ()
{     Bazaar  B ;       //statement 1
      B. disp() ;       //statement  2
}
```

(i) Will statement 1 initialize all the data members for object B with the values given in the function1? (YES OR NO). Justify your answer suggesting the correction(s) to be made in the above code.

Ans: No. The reason is the constructor should be defined under the public visibility label.

(ii) What shall be the possible output when the program gets executed ? (Assuming, if required _ the suggested correction(s) are made in the program).

Ans: Possible Output:

```
Electronic–Calculator:10@225
```

4. Answer the question (i)and (ii)after going through the following class:

```
 class Maths
{     char Chapter[20]
      int Marks;
public:
      Maths()                //Member Function 1
      {   strcpy (Chapter, "Geometry");
          Marks=10;
          cout <<"Chapter Initialised ";
      }
      -Maths()               //Member Functions 2
      {             cout<<"Chapter Over";
      }
 };
```

(i) Name the specific features of class shown by member Function 1 and Member Function 2 in the above example.

Ans: Member function 1 is a (non-parameterized or default) constructor (, which will be executed automatically at the time of creation of an object of class Maths).

Member function 2 is a destructor (,which will be executed automatically at the time of destruction of an object of class Maths).

(ii) How would Member Function 1 and Member Function 2 get executed ?

Ans: They will be executed automatically.
Member function 1 will be executed at the time of creation of an object of class Maths. Member function 2 will be executed at the time of destruction of an object of class Maths.

5.Answer the following questions (i) and (ii) after going through the following class.

```
    class Exam
    {     int Year;
        public:
          Exam(int y)     //Constructor 1
          {         Year=y;
          }
          Exam(Exam &t);
                    //Constructor 2
    };
```

(i)Create an object, such that it invokes Constructor 1

Ans: Exam E((2008);

(ii)Write complete definition for constructor 2.

Ans:
```
Exam(Exam &t)     //Copy Constructor.
  {     Year=t.Year;
  }
```

6. Answer the following questions (i) and (ii) after going through the following class.

```
        class Test
        {    char Paper[20];
             int Marks
        public:
             Test()     //Function 1

             {     strcpy(Paper,"Computer");
                   Marks=0;
             }

           //Function 2
             Test(char P[])
             {     strcpy(Paper,P);
                   Marks=0;
             }

          //Function 3
             Test(int M)
             {     strcpy(Paper,"Computer");
                   Marks=M;
              }
              Test(char P[],int M)
               //Function 4
              {     strcpy(Paper,P);
                   Marks=M;
              }
          };
```

(i)Which feature Object Oriented programming is demonstrated using Function 1, Function 2, Function 3 and Function 4 in the above class text?

Ans: Function overloading (here it is constructor overloading).

(ii)Write statements in C++ that would execute Function 2 and Function 4 of class Text.

Ans: **(let** char name[20];
int X=60;
strcpy(name,"COMPUTERSCIENCE");
are declared in the program)

```
  (i)      Test  A(name);
          //Will execute Funciton 2
  (ii)     Test  B(name,X);
          //Will execute Function 4
```

OUTPUT:

1. Write the **output** of the following program.4

Ans: #include<iostream.h>

```
class Counter
{  private:
      unsigned int count;
   public:
      Counter()
      {     count=0;
      }
      void inc_Count()
      {      count++;
      }
      int get_Count()
      {      return count;
      }
};
void main()
{     Counter C1,C2;
      cout<<"\nC1="<<C1.get_Count();
      cout<<"\nC2="<<C2.get_Count();
      C1.inc_Count();
```

Output:

```
C1=0
C2=0
C1=1
C2=2
```

```
                C2.inc_Count();
                C2.inc_Count();
                cout<<"\nC1="<<C1.get_Count();
                cout<<"\nC2="<<C2.get_Count();
}
```

2. b) Answer the questions (i) and (ii) after going through the following class: 2

```
class Seminar
{           int Time;
    public:
            Seminar() //Function 1
            {   Time=30;
                cout<<"Seminar starts now"<<end1;
             }
            void Lecture() //Function 2
            {
        cout<<"Lectures in the seminar
on"<<end1;
             }
            Seminar(int Duration) //Function 3
            {   Time=Duration;
             cout<<"Seminar starts now"<<end1;
            }
            ~Seminar( )//Function 4
            {   cout<<"Vote of thanks"<<end1;
            }
};
```

i) In Object Oriented Programming, what is Function 4 referred as and when does it get invoked/called?

A) Destructor, it is invoked as soon as the scope of the object gets over. 2

ii) In Object Oriented Programming, which concept is illustrated by Function 1 and Function 3 together? Write an example illustrating the calls for these functions.

A) Constructor Overloading (or Function Overloading or Polymorphism)

```
Seminar S1; //Function 1
Seminar S2(90); //Function 3
```

THEORY QUESTIONS:

1.What is copy constructor?Give ane example in C++ to illustrate copy con-structor.

2.Differentiate between Constructor and Destructor function in context of Classes and Objects Using C++?

3.What is a default constructor? How does it differ from destructor?

4.Differentiate between a default constructer and copy constructer, giving suitable examples of each.

5. Why is destructor function required in classes? Illustrate with the function with an example.

What is a copy constructor? What do you understand by constructer overloading?

" If WEALTH is lost, nothing is lost.
If HEALTH is lost, something is lost.
If CHARACTER is lost, everything is lost."

"The fear of the Lord is the beginning of wisdom"