Unit - IV CHAPTER – 14

Classes and objects in C++

Part – 1
Choose the best answer

1. The variables declared inside the class are known as data members and the functions are known as
(A) data functions (B) inline functions **(C) member functions** (D) attributes

2. Which of the following statements about member functions are True or False?
i) A member function can call another member function directly with using the dot operator.
ii) Member function can access the private data of the class.
(A) i-True, ii-True **(B) i-False, ii-True** (C) i-True, ii-False (D) i-False, ii-False

3. A member function can call another member function directly, without using the dot operator called as
(A) sub function (B) sub member
(C) nesting of member function (D) sibling of member function

4. The member function defined within the class behave like
(A) inline functions (B) Non inline function (C) Outline function (D)Data function

5. Which of the following access specifier protects data from inadvertent modifications?
(A) Private (B) Protected (C) Public (D) Global

6.
```
class x
{
int y;
public:
x(int z){y=z;}
} x1[4];
int main()
{ x x2(10);
return 0;}
```
How many objects are created for the above program
(A) 10 (B) 14 **(C) 5** (D) 2

7. State whether the following statements about the constructor are True or False.
i) constructors should be declared in the private section.
ii) constructors are invoked automatically when the objects are created.
(A) True, True (B) True, False **(C) False, True** (D) False, False

8. Which of the following constructor is executed for the following prototype ?
add display(add &); // add is a class name
(A) Default constructor (B) Parameterized constructor
(C) Copy constructor (D) Non Parameterized constructor

9. What happens when a class with parameterized constructors and having no default constructor is used in a program and we create an object that needs a zero-argument constructor?
(A) Compile-time error (B) Domain error (C) Runtime error (D) Runtime exception.

10. Which of the following create a temporary instance?
(A) Implicit call to the constructor **(B) Explicit call to the constructor**
(C) Implicit call to the destructor (D) Explicit call to the destructor

PART II

Answer to all the questions (2 Marks):

1. What are called members?

- ➢ The class body contains the declaration of its members (Data member and Member functions).

The class body has three access specifiers are,

1. private ***2. public*** ***3. protected***

2. Differentiate structure and class though both are user defined data type.

The only difference between structure and class is the members of structure are by default **public** where as it is **private in class**.

3. What is the difference between the class and object in terms of oop?

Class	Object
A class specification just defines the properties of a class.	The class variables are called Object.
Define using Class-Keyword	Define data-Type.
Ex: Class student	Ex: int no;

4. Why it is considered as a good practice to define a constructor though compiler can automatically generate a constructor?

- ➢ Constructor is a special initialization member function of a class that is called automatically whenever an instance of a class is declared or created.

The main function of the constructor is

1) To allocate memory space to the object and
2) To initialize the data member of the class object

5. Write down the importance of destructor.

- ➢ The purpose of the destructor is to free the resources that the object may have acquired during its lifetime.
- ➢ A destructor function removes the memory of an object which was allocated by the constructor at the time of creating a object.

PART III

Answer to all the questions (3 Marks):

1. Rewrite the following program after removing the syntax errors if any and underline the errors:

```
#include<iostream>
#include<stdio.h>
classmystud
{ intstudid =1001;
char name[20];
public
mystud( )
{ }
void register ( ) {cin>>stdid;gets(name);
}
void display ( )
{ cout<<studid<<”: “<<name<<endl;}
}
int main( )
{ mystud MS;
register.MS( );
MS.display( );
}
```

```
#include<iostream>
#include<stdio.h>
Class mystud
{ int studid;
char name[20];
public:
mystud( )
{
studid =1001;
}
void register ( ) {cin>>stdid;gets(name);
}
void display ( )
{ cout<<studid<<”: “<<name<<endl;}
};
int main( )
{ mystud MS;
MS.register( );
MS.display( );
}
```

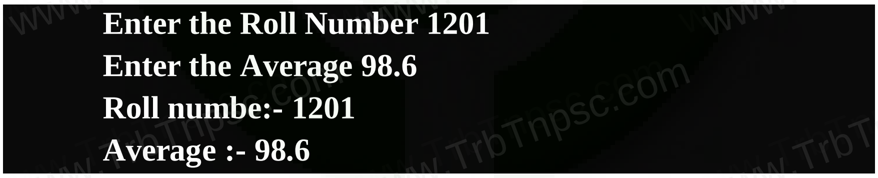
2. Write with example how will you dynamically initialize objects?

- When the initial values are provided during runtime then it is called dynamic initialization.

Example program to illustrate dynamic initialization

```
#include<iostream>
using namespace std;
class X
{
int n;
float avg;
public:
X(int p,float q)
{
n=p;
avg=q;
}
void disp()
{
cout<<"\n Roll numbe:- " <<n;
cout<<"\nAverage :- "<<avg;
}
};
int main()
{
int a ; float b;238
cout<<"\nEnter the Roll Number";
cin>>a;
cout<<"\nEnter the Average";
cin>>b;
X x(a,b); // dynamic initialization
x.disp();
return 0;
}
```

Output:



```
Enter the Roll Number 1201
Enter the Average 98.6
Roll numbe:- 1201
Average :- 98.6
```

3. What are advantages of declaring constructors and destructor under public accessibility?

- A constructor can be defined either in private or public section of a class.
- If it is defined in public section of a class, then its object can be created in any function.
- Easy to access other classes compare to others(private, protected).

4. Given the following C++ code, answer the questions (i) & (ii).

```
class TestMeOut
{
public:
~TestMeOut() //Function 1
{cout<<“Leaving the examination hall”<<endl;}
TestMeOut() //Function 2
{cout<<“Appearing for examination”<<endl;}
void MyWork() //Function 3
{cout<<“Attempting Questions//<<endl;}
};
```

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

Function 1 is called **Destructor**. (Class name same as this function 1 name. Also starting with ~ symbol)

It will be automatically **invoked** when object goes out of scope. (At the end of a program)

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

Function 2 is called Constructor. (Class name same as this function 1 name. But not starting with ~ symbol)

It will be automatically **invoked** when object goes out of scope. (At the end of a program)

5. Write the output of the following C++ program code :

```
#include<iostream>
using namespace std;
class Calci
{
char Grade;
int Bonus;
public:
Calci() {Grade='E'; Bonus=0;} //ascii value of A=65
void Down(int G)
{
Grade-=G;
}
void Up(int G)
{
Grade+=G;
Bonus++;
}
void Show()
{
cout<<Grade<<"#"<<Bonus<<endl;
}
};
int main()
{
Calci c;
c.Down(3);
c.Show();
c.Up(7);
c.Show();
c.Down(2);
c.Show();
return 0;
}
```

Output:

```
B#0

I#1

G#1
```

PART IV

Answer to all the questions (5 Marks):

1. Explain nested class with example.

- When one class become the member of another class then it is called Nested class and the relationship is called containership.

Classes can be nested in two ways.

1. By defining a class within another class
2. By declaring an object of a class as a member to another class

Defining a class with in another

When a class is declared with in another class, the inner class is called as Nested class (inner class) and the outer class is known as Enclosing class. Nested class can be defined in private as well as in the public section of the Enclosing class.

Example:

```
#include<iostream>
using namespace std;
class enclose
{
private:
int x;
class nest
{
private :
int y;
public:
int z;
void prn()
{
y=3;z=2;
cout<<"\n The product of"<<y<<'*'<<z<<"= "<<y*z<<"\n";
}
};          //inner class definition over
nest n1;
public:
nest n2;
void square()
{
n2.prn();     //inner class member function is called by its object
x=2;
n2.z=4;
cout<<"\n The product of " <<n2.z<<'*'<<n2.z<<"= "<<n2.z*n2.z<<"\n";
cout<<"\n The product of " <<x<<'*'<<x<<"= "<<x*x;
}
};        //outer class definition over
int main()
{
enclose e;
e.square();     //outer class member function is called
}
```

Output:

```
The product of 3*2=6
The product of 4*4=16
The product of 2*2=4
```

In the above program the inner class nest is defined inside the outer class enclose. nest is accessed by enclose by creating an object of nest.

2. Mention the differences between constructor and destructor.

CONSTRUCTOR	DESTRUCTOR
The name of the constructor must be same as that of the class.	The Destructor has the same as that of the class prefixed by the Tilde symbol (~).
The constructor function can be overloaded.	The Destructor function can’t be overloaded.
A constructor can have parameter (Arguments) list.	The Destructor cannot have parameter (Arguments) list.
Constructor cannot be inherited. But a derived class can call the base class constructor.	Destructor cannot be inherited.
The constructor is executed automatically when the object is created.	The destructor is executed automatically when the control reaches the end of class scope to destroy the object.
Allocated memory space for the objects.	Destroy the object.

3. Define a class RESORT with the following description in C++:

Private members:

Rno	// Data member to store room number
Name	//Data member to store user name
Charges	//Data member to store per day charge
Days	//Data member to store the number of days
Compute ( )	// A function to calculate total amount as Days * Charges and if the //total amount exceeds 11000 then total amount is 1.02 * Days *Charges

Public member:

getinfo( )	// Function to Read the information like name , room no, charges and days
dispinfo ( )	// Function to display all entered details and total amount calculated //using COMPUTE function

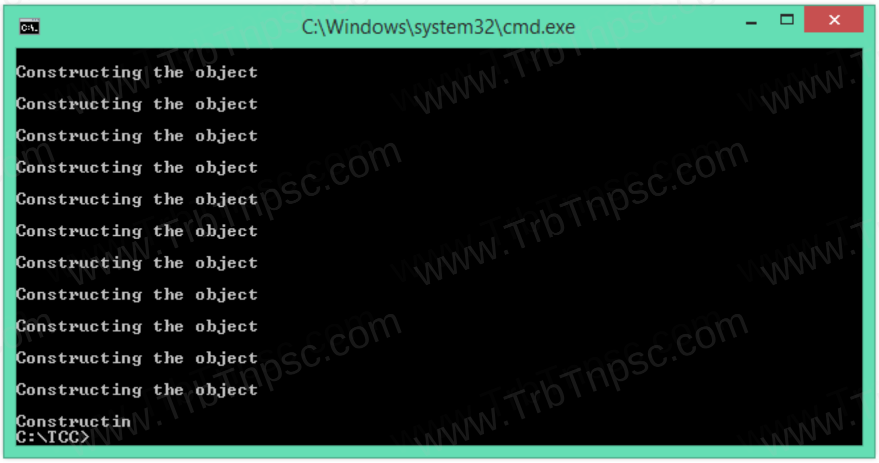
```
#include<iostream>
using namespace std;
class RESORT
{
private:
int Rno, Days, charges;
char Rname[20];
int compute()
{
if ( Days * Charges > 11000 )
return ( Days * Charges * 1.02 );
else
return ( Days * Charges);
}
public:
getinfo()
{
cout<< “\n Enter customer name :”;
cin>>Rname;
cout<< “\n Enter charges per day:”;
cin>>Charges;
```

```
cout<< “\n Enter Number of days:”;
cin>>Days;
cout<< “\n Enter Room Number:”;
cin>>Rno;
}
dispinfo()
{
cout<< “\n Room Number:” <<Rno;
cout<< “\n Customer name:” <<Rname;
cout<< “\n Charges per day:” <<Charges;
cout<< “\n Number of days:” <<Days;
cout<< “\n Total Amount:” <<compute();
}
};
Int main()
{
RESORT S;
S.getinfo();
S.dispinfo();
}
```

4. Write the output of the following

```
#include<iostream>
#include<stdio.h>
using namespace std;
class sub
{
int day, subno;
public :
sub(int,int); // prototype
void printsub()
{ cout<<" subject number : "<<subno;
cout<<" Days : " <<day;
}
};
sub::sub(int d=150,int sn=12)
{ cout<<endl<<"Constructing the object "<<endl;
day=d;
sub no=sn;
}
class stud
{
int rno;
float marks;
public:
stud( )
{ cout<<"Constructing the object of students "<<endl;
rno=0;
marks=0.0;
}
void getval()
{
cout<<"Enter the roll number and the marks secured ";
cin>>rno>>marks;
}
void printdet()
{ cout<<"Roll no : "<<rno<<"Marks : "<<marks<<endl;
}
};
class addmission {
sub obj;
stud objone;
float fees;
public :
add mission ( )
{ cout<< "Constructing the object of admission "<<endl;
fees=0.0;
}
void printdet( )
{ objone.printdet();
obj.printsub( );
cout<<"fees : "<<fees<<endl ;
}
};
int main()
{system("cls");
addmission adm;
cout<<endl<< "Back in main ( )";
return 0; }
```

Output:

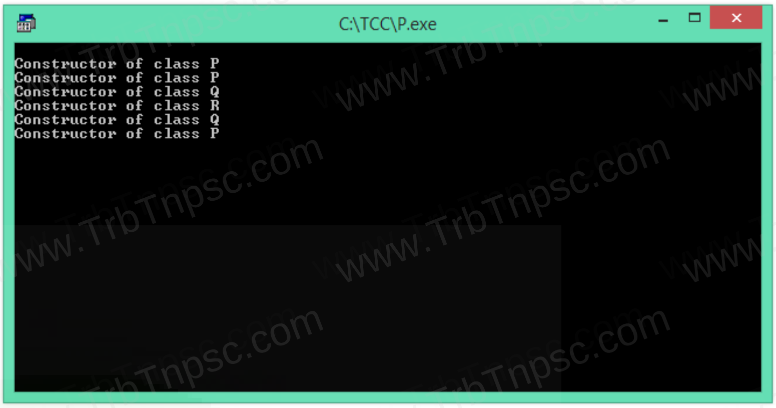


```
Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructing the object

Constructin
C:\TCC>
```

5. Write the output of the following

```
#include<iostream>
#include<stdio.h>
using namespace std;
class P
{ public:
P ( )
{ cout<< "\nConstructor of class P "; }
~ P ( )
{ cout<< "\nDestructor of class P "; }
};
class Q
{ public:
Q( )
{ cout<<"\nConstructor of class Q "; }
~ Q( )
{ cout<< "\nDestructor of class Q "; }
};
class R
{ P obj1, obj2;
Q obj3;
public:
R ( )
{ cout<< "\nConstructor of class R ";}
~ R ( )
{ cout<< "\nDestructor of class R ";}
};
int main ( )
{
Ro R;
Q oq;
P op;
return 0;
}
```

Output:



```
Constructor of class P
Constructor of class P
Constructor of class Q
Constructor of class R
Constructor of class Q
Constructor of class P
```

Unit - IV CHAPTER – 15

POLYMORPHISM

PART I

Choose the correct answers

1. Which of the following refers to a function having more than one distinct meaning?

(A) Function Overloading (B) Member overloading
(C) Operator overloading (D) Operations overloading

2. Which of the following reduces the number of comparisons in a program ?

(A) Operator overloading (B) Operations overloading
(C) Function Overloading (D) Member overloading

3. void dispchar(char ch='$',int size=10)

```
{
for(int i=1;i<=size;i++)
cout<<ch;
}
```

How will you invoke the function dispchar() for the following input? To print $ for 10 times

(A) dispchar(); (B) dispchar(ch,size); (C) dispchar($,10); (D)dispchar('$',10 times);

4. Which of the following is not true with respect to function overloading?

(A) The overloaded functions must differ in their signature.
(B) The return type is also considered for overloading a function.
(C) The default arguments of overloaded functions are notconsidered for Overloading.
(D) Destructor function cannot be overloaded.

5. Which of the following is invalid prototype for function overloading

(A) void fun (intx); void fun (char ch) ; **(B) void fun (intx); void fun (inty);**
(C) void fun (double d); void fun (char ch); (D) void fun (double d); void fun (inty);

6. Which of the following function(s) combination cannot be considered as overloaded function(s) in the given snippet ?

```
void print(char A,int B); // F1
void printprint(int A, float B); // F2
void Print(int P=10); // F 3
void print(); // F4
```

(A) F1,F2,F3,F4 **(B) F1,F2,F3** (C) F1,F2,F4 (D) F1,F3,F4

7. Which of the following operator is by default overloaded by the compiler?

(A) * (B) + **(C) + =** (D) = =

Based on the following program answer the questions (8) to (10)

```
#include<iostream>
using namespace std;
class Point {
private:
int x, y;
public:
Point(int x1,int y1)
{
x=x1;y=y1;
}
void operator+(Point &pt3);
void show() {cout << "x = " << x << ", y = " << y; }
};
void Point::operator+(Point &pt3)
{
x += pt3.x;
y += pt3.y;
}
int main()
{
Point pt1(3,2),pt2(5,4);
pt1+pt2;
pt1.show();
return 0;
}
```

8. Which of the following operator is overloaded?

(A) + (B) operator (C) :: (D) =

9. Which of the following statement invoke operator overloading?

(A) pt1+pt2; (B) Point pt1(3,2),pt2(5,4); (C) pt1.show(); (D) return 0;

10. What is the output for the above program?

(A) x=8, y=6 (B) x=14, y=14 (C) x=8, y=6 (D) = x=5, y=9

PART II

Answer to all the questions (2 Marks):

1. What is function overloading?

- The ability of the function to process the message or data in more than one form is called as function overloading.
- In other words function overloading means two or more functions in the same scope share the same name but their parameters are different. In this situation, the functions that share the same name are said to be overloaded and the process is called function overloading.

2. List the operators that cannot be overloaded.

- Scope operator **(::)**
- Sizeof
- Member selector **(.)**
- Member pointer selector **(*)**
- Ternary operator **(?:)**

3. class add{int x; public: add(int)}; Write an outline definition for the constructor.

```
add : : add(int a)
{
x = a ;
cout<< "\n parameterized constructor";
}
```

4. Does the return type of a function help in overloading a function?

No.

- The return type of a function does not help in overloading a function.
- Only arguments are considered.

5. What is the use of overloading a function?

- Function overloading is not only implementing polymorphism but also reduces the number of comparisons in a program and makes the program to execute faster.
- Program complexity is reduced.
- It also helps the programmer by reducing the number of function names to be remembered.

PART III

Answer to all the questions (3 Marks):

1. **What are the rules for function overloading?**
   - The overloaded function must differ in the number of its arguments or data types.
   - The return type of overloaded functions are not considered for overloading same data type.
   - The default arguments of overloaded functions are not considered as part of the parameter list in function overloading.
2. **How does a compiler decide as to which function should be invoked when there are many functions? Give an example.**

The number and types of a function's parameters are called the **function's signature.** When you call an overloaded function, the compiler determines the most appropriate definition to use, by comparing the argument types you have used to call the function with the parameter types specified in the definitions. The process of selecting the most appropriate overloaded function or operator is called **overload resolution.**

EX:

```
float area ( float radius);
float area ( float half, float base, float height );
float area ( float length , float breadth);
```

3. **What is operator overloading? Give some example of operators which can be overloaded.**
   - The term operator overloading, refers to giving additional functionality to the normal C++ operators like +,++,-,—,+=,-=,*.<,>.
   - It is also a type of polymorphism in which an operator is overloaded to give user defined meaning to it.
4. **Discuss the benefit of constructor overloading?**
   - Function overloading can be applied for constructors, as constructors are special functions of classes.
   - A class can have more than one constructor with different signature.
   - Constructor overloading provides flexibility of creating multiple type of objects for a class.

5. class sale (int cost, discount ;public: sale(sale &); Write a non inline definition for constructor specified;

```
sale : : ( sale &s )
{
cost = s.cost;
discount = s.discount;
}
```

PART IV

Answer to all the questions (5 Marks):

1. What are the rules for operator overloading?

- Precedence and Associativity of an operator cannot be changed.
- No new operators can be created, only existing operators can be overloaded.
- Cannot redefine the meaning of an operator's procedure. You cannot change how integers are added. Only additional functions can be to an operator
- Overloaded operators cannot have default arguments.
- When binary operators are overloaded, the left hand object must be an object of the relevant class

2. Answer the question (i) to (v) after going through the following class.

```
classBook
{
intBookCode ; char Bookname[20];float fees;
public:
Book( ) //Function 1
{
fees=1000;
BookCode=1;
strcpy (Bookname,"C++");
}
void display(float C) //Function 2
{
cout<<BookCode<<":"<<Bookname<<":"<<fees<<endl;
}
~Book( ) //Function 3
{
cout<<"End of Book Object"<<endl;
}
Book (int SC, char S[ ], float F) ; //Function 4
};
```

(i) In the above program, what are Function 1 and Function 4 combined together referred as?

Constructor

(ii) Which concept is illustrated by Function3? When is this function called/ invoked?

Destructor. Executed automatically when object goes out of scope.

(iii) What is the use of Function3?

To remove the memory space of the object allocated at the time of creation.

(iv) Write the statements in main to invoke function1 and function 2 .

Function 1 invoke → **Book()** constructor function automatically when object b Created.
Function 2 invoke → **display(float C)** function passing a float value.

(v) Write the definition for Function4.

Book (int SC, char S[ ], float F)

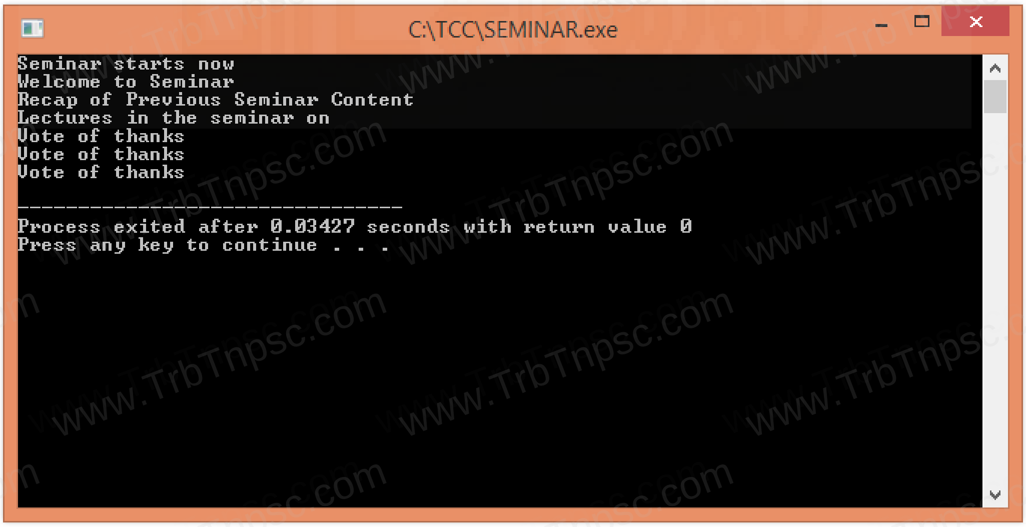
SC	-	**Bookcode**
S	-	**Bookname**
F	-	**Fees**

```
fees=1000;
BookCode=1;
strcpy (Bookname,"C++");
```

3. Write the output of the following program

```
#include<iostream>
using namespace std;
class Seminar
{
int Time;
public:
Seminar()
{
Time=30;cout<<"Seminar starts now"<<endl;
}
void Lecture()
{
cout<<"Lectures in the seminar on"<<endl;
}
Seminar(int Duration)
{
Time=Duration;cout<<"Welcome to Seminar "<<endl;
}
Seminar(Seminar &D)
{
Time=D.Time;cout<<"Recap of Previous Seminar Content "<<endl;
}
~Seminar()
{
cout<<"Vote of thanks"<<endl;
}
};
int main()
{
Seminar s1,s2(2),s3(s2);
s1.Lecture();
return 0;
}
```

Output:

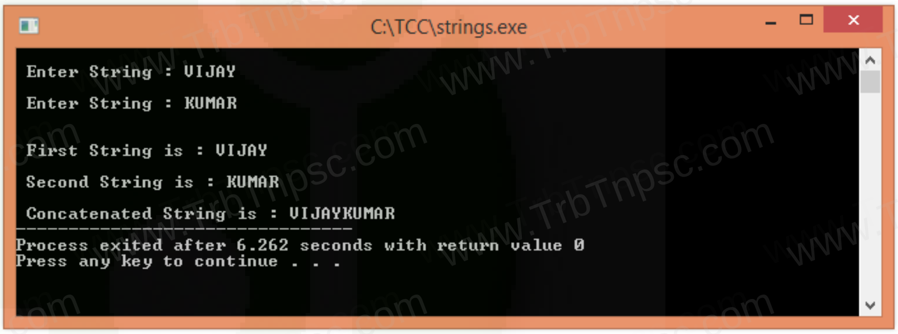


```
Seminar starts now
Welcome to Seminar
Recap of Previous Seminar Content
Lectures in the seminar on
Vote of thanks
Vote of thanks
Vote of thanks

--------------------------------
Process exited after 0.03427 seconds with return value 0
Press any key to continue . . .
```

4. Debug the following program

```
#include<iostream>
using namespace std;
class String
{
public:
charstr[20];
public:
void accept_string
{
cout<<"\n Enter String : ";
cin>>str;
}
display_string()
{
cout<<str;
}
String operator *(String x) //Concatenating String
{
String s;
strcat(str,str);
strcpy(s.str,str);
goto s;
}
}
int main()
{
String str1, str2, str3;
str1.accept_string();
str2.accept_string();
cout<<"\n\n First String is : ";
str1=display_string();
cout<<"\n\n Second String is : ";
str2.display_string();
str3=str1+str2;
cout>>"\n\n Concatenated String is : ";
str3.display_string();
return 0;
}
```

Output:



```
Enter String : VIJAY
Enter String : KUMAR

First String is : VIJAY
Second String is : KUMAR
Concatenated String is : VIJAYKUMAR
--------------------------------
Process exited after 6.262 seconds with return value 0
Press any key to continue . . .
```

S.No	Line No	Error	Correction
1	2	String header file missing	#includestring.h>
2	6	charstr[20];	char str[20];
3	8	void accept_string	void accept_string()
4	13	display_string()	void display_string()
5	17	String operator *(String x)	String operator +(String x)
6	20	strcat(str, str);	strcat(str, x.str);
7	22	goto s;	return(s);
8	24	}	};
9	31	str1=display_string();	str1.display_string();
10	35	cout>>"\n\n Concatenated String is : ";	cout<<"\n\n Concatenated String is : ";

5. Answer the questions based on the following program

```
#include<iostream>
#include<string.h>
using namespace std;
class comp {
public:
chars[10];
void getstring(char str[10])
{
strcpy(s,str);
}
void operator==(comp);
};
void comp::operator==(comp ob)
{
if(strcmp(s,ob.s)==0)
cout<<"\nStrings are Equal";
else
cout<<"\nStrings are not Equal";
}
int main()
{
comp ob, ob1;
char string1[10], string2[10];
cout<<"Enter First String:";
cin>>string1;
ob.getstring(string1);
cout<<"\nEnter Second String:";
cin>>string2;
ob1.getstring(string2);
ob==ob1;
return 0;
}
```

(i) Mention the objects which will have the scope till the end of the program.

ob and ob1

(ii) Name the object which gets destroyed in between the program.

ob

(iii) Name the operator which is over loaded and write the statement that invokes it.

Operator overloaded is: = =
Invoke the statement is: **ob = = ob1**

(iv) Write out the prototype of the overloaded member function.

void comp :: operator = = (comp ob)

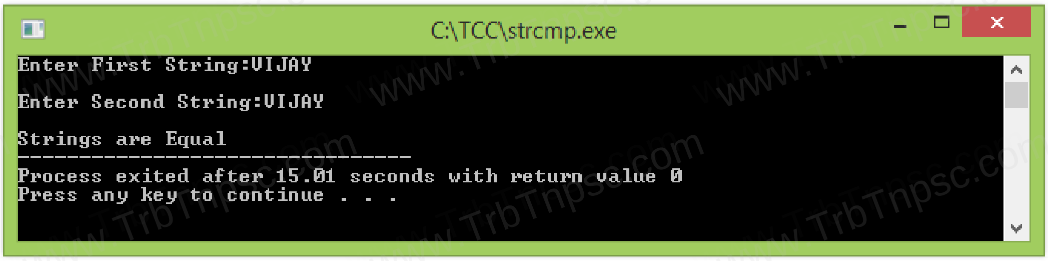
v) What types of operands are used for the overloaded operator?

User defined

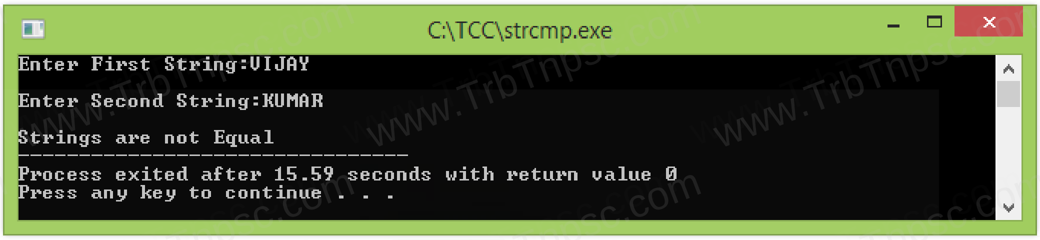
(vi) Which constructor will get executed?

Constructor not used in this program. (Only default constructor to be executed)

(vii) Write the output of the program.



```
Enter First String:VIJAY

Enter Second String:VIJAY

Strings are Equal
--------------------------------
Process exited after 15.01 seconds with return value 0
Press any key to continue . . .
```



```
Enter First String:VIJAY

Enter Second String:KUMAR

Strings are not Equal
--------------------------------
Process exited after 15.59 seconds with return value 0
Press any key to continue . . .
```