Assignment 1 Set-C

Course Code :- CAP444

Date of Submission:-

Name :- Sirjanpreet Kaur

Roll no :- B56

Section :- D2112

Registration no :- 12107974

Name – Sirjanpreet Kaur
Reg. no - 12107974
Roll no - B56

Assignment 1 Set-C

Ques 1: What are the different types of comparison and logical operators are supported in C++? Differentiate between comparison operators and logical operators.

Ans: An operator is a special symbol that tells the compiler to perform specific mathematical or logical operation on one or more operands where an operand can be a variable, constant or an expression.

There are seven types of operators in C++. One of the most common operators are given below:-

1. **Comparison Operators:-** Comparison operators are also known as relational operators are used to make comparisons between two operands. Comparison operators require two operands. The result of comparison operation is a boolean value that can only be true (non-zero) or false (zero) according to result of the comparison. There are six types of comparison operators:-

Operator Symbol	Operation Performed	Use	Result
==	Equal to	x == y	True if x is equal to y else false.
!=	Not equal to	x != y	True if x is not equal to y else false

Name - Sanjeet Kaur
Reg. No - 12107974
Roll no - B56

Operator Symbol	Operation Performed	Use	Result
>	Greater than	x > y	True if x is greater than y else false
<	Less than	x < y	True if x is less than y else false
>=	Greater than or equal to	x >= y	True if x is greater than or equal to y else false
<=	Less than or equal to	x <= y	True if x is less than or equal to y else fals.

for example:- If radius = 0, num = 5 and ch = 'y', where radius and num are integer type variable and ch is character type variable then

```
radius < 0 // will return false
ch == 'y' // will return true
n%2 == 1 // will return true
```

2. Logical Operators:- Logical operators are used to combine one or more relational expressions that results in formation of complex expressions known as Logical Operators. Logical operators evaluate the result of logical expression in boolean values that can only be True or false according to the result of the logical expression.

There are three types of Logical operators:-

1. Logical AND
2. Logical OR
3. Logical NOT

Name - Sirjanpreet Kaur
Reg. No - 12107974
Roll no. - B56

Operator Symbol	Operation Performed	Description	Example
&&	Logical AND	Returns true if both statements are true else false	x<2 && x<5
||	Logical OR	Returns true if one of the statements is true else false	x<5 || x<4
!	Logical NOT	Reverse the result, returns false if the result is true	!(x<5 && x<10)

For example:- The value of x is 10 and y is 20.

(i) ((x>30) && (y>5)) || (10>30) && (20>5)

returns False because it is having AND operator. So, both the conditions must be true.

(ii) ((x>30) || (y>5)) || (10>30) || (20>5)

return true as one of the statements is true. In logical OR, if one statement is true, then we don't need to check next statement as only checking of one statement is sufficient.

(iii) (!(x>3) || (y>5)) || (!(10>3) || (20>5))

Returns False as both the statements are true. But due to operator Logical NOT, it reverses the result. So it returns false.

Name - Sanjanpreet kaur
Reg. No - 12107974
Roll no - B56

Difference :-

	Comparison Operators	Logical Operators
1.	Comparison operators are used to compare two operands.	Logical operators are used to combine one or more relational expression.
2.	The result of comparison operators is in the form of Boolean either true or false	The result of Logical operators are also in the form of Boolean either true or false
3.	There are more than six types of comparison operators	There are only three types of operators
4.	The operators are:- <, >, <=, >=, == and !=	The operators are :- && - Logical AND, || - Logical OR and ! - Logical NOT
5.	Comparison operators compare two values and produce Boolean Result	Logical operators combine Boolean values and produce Boolean result.

Name - Satyampreet Kaur
Reg. No - 12107974
Roll No - B56

6. Comparison operator can have any operand values and return boolean	Logical operators always have boolean operands and return boolean

Ex. Example:
#include<iostream.h>
#include<stdio.h>
int x=10;
int y=28;
if(x=y)
cout<<"Equal";
if(x>y)
cout<<"x is greater than y";
if(x<y)
cout<<"x is less than y";
if(x!=y)
cout<<"x is not equal to y";
if(x<=y)
cout<<"x is less than or equal to y";
if(x>=y)
cout<<"x is greater than or equal to y";
return 0;
}.
	Example:
	#include<iostream.h>
	#include<stdio.h>
	int x=10;
	int y=28;
	int a=18;
	int b=20;
	if(x<y && a>=b)
	cout<<"x is less than y AND a is equal to b\n";
	if(x<y || a>=b)
	cout<<"x is less than y OR a is equal to b\n");
	if(!x)
	cout<<"x is zero\n");
	return 0;
	}

Output: x is less than y x is not equal to y x is lesser or equal to y.	Output: x is less than y OR a is equal to b

Name- Sirjanpreet Kaur
Reg. No- 12107974
Roll. No- B56

Ques2:- Create a class named 'Time' with data members (hours, mins, secs). Derive from this class to some new classes named 'hourConvertor', 'minConv-ertor' with data members total_hours, total_mins and total_secs. With the help of appropriate function calculate total hours, mins and secs for any integer value (in days).

Ans

```
#include<iostream>
using namespace std;

class Time
{
   public:
     int hours, mins, secs;
};

class hourConvertor : public Time
{
   public:
     int total_hours, days;
     void hourData()
     {
       cout<<"Enter any value: ";
       cin>>days;
       total_hours = days*24;
       cout<<"Total_hours: "<<total_hours<<endl;
     }
};

class minConvertor : public hourConvertor
{
```

Name - Sirjanpreet kaur
Reg. No. - 12107774
Roll. No. - B56

```
public:
int total_mins;
void minData()
{
    total_mins = total_hours * 60;
    cout << "Total_mins : " << total_mins << endl;
}
};

class secConvertor : public minConvertor
{
    public:
    int total_secs;
    void secData()
    {
        total_secs = total_mins * 60;
        cout << "Total_secs : " << total_secs << endl;
    }
};

int main()
{
    secConvertor s;
    s.hourData();
    s.minData();
    s.secData();
    return 0;
}
```

Name - Sirjanpreet Kaur
Reg. No - 12107974
Roll. No - B56

Output:-

```
Enter any value : 4
Total_hours : 96
Total_mins : 5760
Total_secs : 345600
```

Ques3:- Why constructor is required? Can we overload constructor in C++? Discuss with the help of example.

Ans:- Constructor is a special member function of a class which is used to create and initialize the objects.

- Constructor is a member function of a class.
- The constructor has the same name as that of class
- It has no return type, so we can't use return keyword.
- Constructor is implicitly invoked when an object is created
- Constructor is used to solve the problem of initialization.

Name-Simranpreet Kaur
Reg. No- 12107974
Role No- B56

Syntax :-

```
class CLASSNAME
{
  public:
    CLASSNAME([parameter list]) //constructor
    {                           //definition
      ---
    }
};
```

Each time when an object is created, a constructor is invoked. But when we define objects and classes without defining any constructor for a class, then the compiler automatically generates a constructor of its own with no parameters i.e default Constructor. That compiler generated default constructor is invoked automatically whenever any object of the class is created but doesn't perform any initialization. But when you will define constructor explicitly, the compiler no longer generates a default constructor.

Constructors are also used to allocate memory at run time using the new operator. It is used to initialize values to the object at the time of its creation in the same way as we initialize values to a variable of built-in types. So, Constructor enables an object to initialize itself at the time of its creation without need to make separate calls.

Name - Sirjanpreet Kaur
Reg No - 12107974
Roll No. - B56

Yes, we can overload constructor in C++. Constructor Overloading allows a class to have more than one constructors that have same name as that of the class but differs only in terms of number of parameters. By overloading a constructor of a class, we make the class more versatile as it allows us to construct objects in a variety of ways.

- Overloaded Constructors have the same name as that of class but differ by number of arguments
- A constructor is called depending upon the number and type of arguments passed.
- While creating object, arguments must be passed to let compiler know which constructor needs to be called.

For example:-

```
#include <iostream>
using namespace std;

class rectangle
{
    private:
        int length, breadth;
    public:
        rectangle()
        {
            length = 0;
            breadth = 0;
            cout<<"constructor with zero parameter called\n";
        }
```

Name- Sirjanpreet Kaur
Reg. No- 12107974
Roll No- B56

```
rectangle(int a)
{
    length = breadth = 0;
    cout << "constructor with one parameter called\n";
}

rectangle(int a, int b)
{
    length = a;
    breadth = b;
    cout << "constructor with two parameters called\n";
}
int area()
{
    return(length * breadth);
}
};
int main()
{
    rectangle r1;
    rectangle r2(5);
    rectangle r3(7,8);
    cout << "\n Area of first rectangle = " << r1.area();
    cout << "\n Area of square = " << r2.area();
    cout << "\n Area of second rectangle = " << r3.area();
    return 0;
}
```

Name - Sirjanpreet Kaur
Reg. No. - 12107974
Roll. No. - B56

Output:-

```
Constructor with zero parameter called
constructor with one parameter called
constructor with two parameters called
Area of first rectangle = 0
Area of Square = 25
Area of second rectangle = 56
```

In this example, class rectangle has three constructors which differs in number of parameters and hence overloaded. On execution :-

(i) The first statement rectangle r1; in main(), an object r1 is created and constructor with no parameter is called as it does not contain any parameter.

(ii) Statement rectangle r2(5); creates object r2 and pass one value. So, it will call single parameter constructor

(iii) Statement rectangle r3(7,8), creates an object r3 and two arguments are specified. So it will call constructor with two parameters.

(iv) The values passed in constructors are used to calculate the area of three rectangles represented by objects r1, r2 and r3.