S. Y. B. Tech. Mechanical
Computer Programming C++ Lab

Experiment No. 1
Programmes on CLASSES AND OBJECTS 01

OBJECTIVE:- To write a program in C++ using class.
THEORY:- Define class and object, write syntax of the class, ways to create objects.

Programme: 5.1 from Balagurusamy Book

Experiment No. 2
Programmes on CLASSES AND OBJECTS 02

OBJECTIVE:- To write programs in C++ using classes

Programme: 5.6 and 5.7 from Balagurusamy Book and any two programmes from exercise

Experiment No. 3
Programmes on Friend Function

OBJECTIVE:- To write a program in C++ using friend functions.
THEORY:- Define the friend function and its advantages

Programme: Programme no 2 and 6 from exercise

Experiment No. 4
Programmes on Function Overloading

OBJECTIVE:- To write a program in C++ to implement function overloading.
THEORY:- Define function volume to calculate volume of three different objects. Name of the function is same but its parameters(number and data types) are different.

ALGORITHM :-

1. Declare function volume by passing arguments of different data types like:
   int volume(int)
   double volume(double, int)
   long volume(long,int,int)

2. In main function call all three functions like:
   volume(10);
   volume(2.5,8);
   volume(100,75,15);

3. Define all three functions.

Sample Input/Output:

volume1: 1000
volume2: 157.26
volume3: 112500

Experiment No. 5
Programmes on Constructors and Destructors

OBJECTIVE: Write a program in C++ to show execution of Constructor and Destructor

THEORY:- Definition of Constructor and Destructor

Programme: From 6.7 from Balagurusamy Book

Experiment No. 6
Programmes on Operator Overloading

OBJECTIVE:- Write a program in C++ to implement operator overloading.

THEORY:- It's common to define a meaning for an existing operator for objects of a new class. Operators are defined as either member functions or friend functions. *Don't use operator overloading* just because it can be done and is a clever trick. The purpose of operator overloading is to make Programmes clearer by using conventional meanings for ==, [], ++, - -etc.

For example, overloading the [] operator for a data structure allows x = v[25] in place of a function call. This is purely a convenience to the user of a class. Operator overloading isn't strictly necessary unless other classes or functions expect operators to be defined (as is sometimes the case).

ALGORITHM :-

1. Define operator function in any class like:
   Class class_name{

```
Void operator ++( )
 {
    --
 }
};
```

2. In main function create object of that particular class.
3. Assign value to any getdata function.
4. call operator like obj_name operator
5. Use display function.

Sample Input/Output:

Value : 7
Value after calling ++ operator: 8
Value : 7
Value after calling -- operator: 6

Experiment No. 7
Programmes on Inheritance 01

OBJECTIVE:- To write a program for implementing following types of inheritance
i) Multilevel inheritance ii) Multiple inheritance

Programmes from Balagurusamy Book with output

Experiment No. 8
Programmes on Inheritance 02

OBJECTIVE:- To write a program for implementing following types of inheritance
i) Hierarchical inheritance ii) Hybrid inheritance

Programmes from Balagurusamy Book with output

Experiment No. 9

Programmes on File Operations

Objective:- To write Programmes in C++ to open a single and multiple files to write the data.

Programmes: 11.1 Working with single file and 11.2 Working with multiple files

Experiment No. 10

Programmes on Graphics

OBJECTIVE:- To write a program in C++ to draw a rectangle of given specification

```
//rectline.cpp
//rectangle and line
#include<graphics.h>
#include<conio.h>
const int W = 75;     //1/2 width of rectangle

class rect
 {
                int  xCo, yCo;   //coordinates of centre
                int  linecolor;   //color of outline
                int  fillcolor;     //color of interior
        public:
                rect()              //no argument constructor
                  { xCo=0;  yCo=0;  linecolor=WHITE; fillcolor=WHITE;}


void set(int x, int y, int lc, int fc)
        {
          xCo=x; yCo=y; linecolor=lc; fillcolor=fc;
        }

void draw()
        {
```

```
            setcolor(linecolor);  //line color setting
            setlinestyle(SOLID_LINE, 0, THICK_WIDTH);
            rectangle(xCo-W, yC0-W, xCo+W, yCo+W);
            setfillstyle(SOLID_FILL, fillcolor);
            floodfill(xCo, yCo, linecolor);
            line(xCo-W, yCo+W, xCo+W, yCo-W);
          }
};


void main()
{
 int driver, mode;
            driver DETECT; //set to best Graphics Mode
            initgraph(&driver, &mode, "\\tc\bgi);
            rect r1;
            r1.set(80, 150, YELLOW, RED); //set position & colors
            r1.draw();       //draw rectangle
            getch();
            closegraph();
  }
```

Output: