EXPERIMENT NO-6

Title: Write a program to implement concept of function overloading.

Objectives:

1. To learn function overloading

Key Concept: Function overloading

Theory:

In a C++ program, several functions can have the same name. This is called function overloading, or overloading a function name. Before we state the rules to overloading a function, let us define the following:

Two functions are said to have different formal parameter lists if both functions have:

- A different number of formal parameters or
- If the number of formal parameters is the same, then the data type of the formal parameters, in the order you list them, must differ in at least one position.

For example, consider the following function declarations:

```
void functionOne(int x)
void functionTwo(int x, double y)
void functionThree(double y, int x)
```

These functions all have different formal parameter lists.

To overload a function name, any two definitions of the function must have different formal parameter lists.

Function overloading: Creating several functions with the same name. The signature of a function consists of the function name and its formal parameter list. Two functions have different signatures if they have either different names or different formal parameter lists. (Note that the signature of a function does not include the return type of the function.) The following functions are overloaded as the function name is same, but the signature is different.

```
int calc_mode( int i_arr[], int size);
float calc_mode( float f_arr[], int size);
double calc_mode (double d_arr[], int size);
```

If a function's name is overloaded, then all of the functions in the set have the same name.

Therefore, all of the functions in the set have different signatures if they have different formal parameter lists.

If a function is overloaded, then in a call to that function, the signature—that is, the formal parameter list of the function—determines which function to execute.

Function overloading is used when you have the same action for different sets of data. For function overloading to work, you must give the definition of each function.

Problem Statement:

1. Write a program to implement Circle class

Program Analysis:

For Circle Class

Properties(Member variables):

centerX, centerY, radius, circumference, area - declare all variables private

Member functions:

constructors: supply following constructors

No argument constructor: Circle()

overloaded constructors: Circle(in cx, int cy) - constructor having arguments for center and default initialization for radius.

Circle(int cx, int cy, int r) - constructor having arguments for center and radius.

setters and getters for center,radius.

calcArea()

getArea()

calcCircumference()

getCircumference()

overloaded functions to check whether a given line is tangent to the circle:

isTangent(int x1, int x2, int y1, int y2) - line specified by two points (x1,y1), (x2,y2)

isTangent(int slope, int intercept) - Line specified by slope and intercept

isTangent(Line tl) - Line specified by line object.

a simple draw function - just print drawing circle draw

A Line class having following attributes and functions

Member variables:

x1, y1, x2, y2, slope, intercept

member functions:

constructor - parameterized and non-parameterized

getters and setters for all member variables

function to calculate slope and intercept

Draw the UML class Diagram for Circle and Line class.