**Chapter 7: User-Defined Functions II**

**Extra Programming Examples**

**Laboratory Exercises (3)**

**Part 1: An Example of Value Parameters**

A Program that illustrates how a value parameter works.

#include<iostream>

using namespace std;

void funcValueParam(int num);

int main()

{

int number = 6;

cout << "Line 2: Before calling the function " << "funcValueParam, number = " << number << endl;

funcValueParam(number);

cout << "Line 4: After calling the function " << "funcValueParam, number = " << number << endl; return 0;

}

void funcValueParam(int num)

{

cout << "Line 5: In the function funcValueParam, " << "before changing, num = " << num << endl;

num = 15;

cout << "Line 7: In the function funcValueParam, "<< "after changing, num = " << num << endl;

}

**Part 2: Examples of Reference Parameters**

**Example 1:** A Program that illustrates how a reference parameter works. This program calculates the factorial of a number.

#include <iostream>

using namespace std;

void getNumber(int &num)

{

cout<<"Enter a number: "<<endl;

cin>>num;

}

void calculateFactorial(int n)

{

int i, fact = 1;

for (i=1; i<=n; i++)

{

fact = fact \* i;

}

cout<<"The factorial of "<<n<<" is: "<<fact;

}

int main()

{

int a;

getNumber(a);

calculateFactorial(a);

return 0;

}

**Example 2:** A Program that illustrates how a reference parameter works. This program swaps two numbers using call by reference

#include <iostream>

using namespace std;

void swapNums(int &x, int &y)

{

int z = x;

x = y;

y = z;

}

int main()

{

int firstNum = 10;

int secondNum = 20;

cout << "Before swap: " << "\n";

cout << firstNum << secondNum << "\n";

// Call the function. will change the values of firstNum and secondNum

swapNums(firstNum, secondNum);

cout << "After swap: " << "\n";

cout << firstNum << secondNum << "\n";

return 0;

}

**Part 3: Function Overloading**

A program that reads two integers and two characters. Overload the function larger to be able to compare the two integers and also compare the two characters. Use the following prototypes:

int larger(int x, int y);

char larger(char first, char second);

#include<iostream>

using namespace std;

int larger(int x, int y);

char larger(char first, char second);

int main()

{

int a,b;

char c,d;

cout<< "Enter two integers\n";

cin>>a>>b;

cout<<"The larger number is: "<<larger(a,b)<<endl<<endl;

cout<<"Enter two characters\n";

cin>>c>>d;

cout<<"The larger character is: "<<larger(c,d)<<endl<<endl;

}

int larger(int x, int y)

{

if(x > y)

return x;

return y;

}

char larger(char x, char y)

{

if(x > y)

return x;

return y;

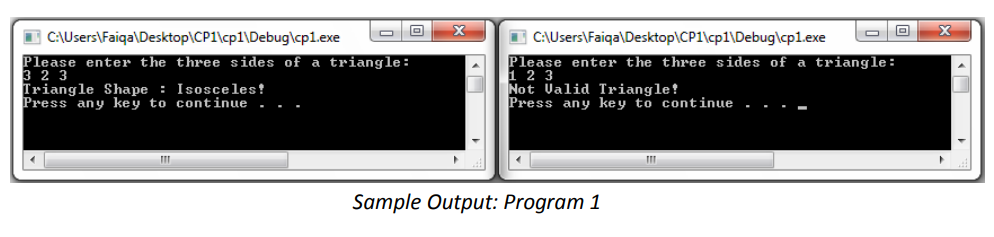
}

**EXERCISES**

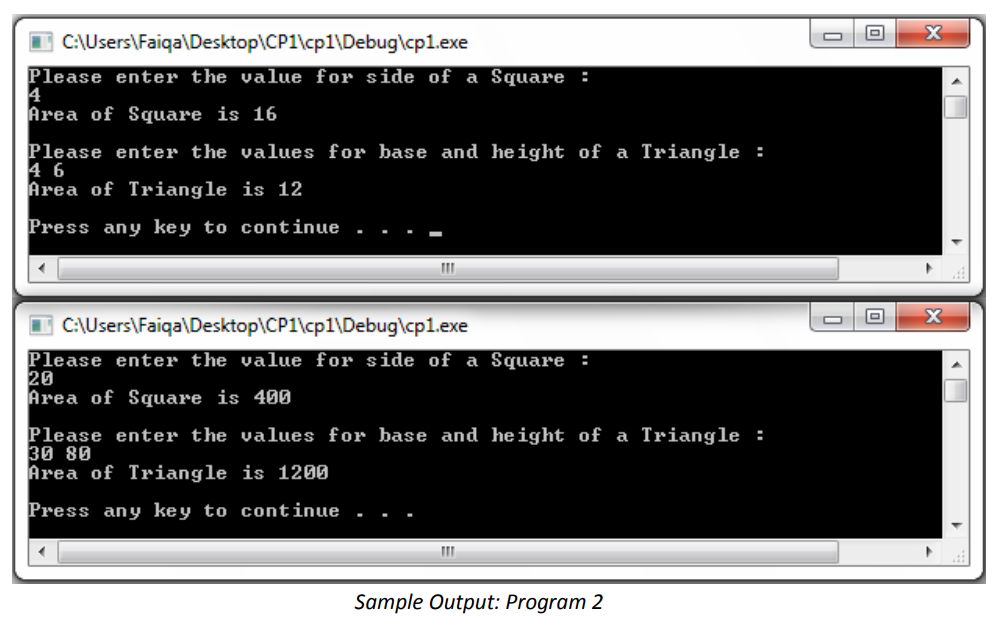
**Enumerations and Function Overloading**

**KEYWORDS: enum, function overloading, default parameter**

**Program 1:** Define an enumeration type, triangleType, that has the values scalene, isosceles, equilateral, and noTriangle. Write a function, triangleShape, that takes as parameters three numbers, each of which represents the length of the sides of the triangle. The function should return the shape of the triangle. (Note: In a triangle, the sum of the lengths of any two sides is greater than the length of the third side).



**Program 2:** Write overloaded function for calculating the area of square and rectangle named area (int side) and area (int base, int height) respectively. Write a program that accept side of a square as input and returns its area using the overloaded function named area. The program also accepts base and height of a rectangle and returns its area using the overloaded function named area.



**Program 3:** Write functions with default argument for addition, multiplication, and division of two numbers. For addition and multiplication, the default values for two numbers will be 0. For division the default value for first and second arguments will be zero and one respectively. Use the above function for finding addition, multiplication, and division when

1) No arguments are supplied,

2) First argument is 2 and second argument is not supplied,

3) Arguments are 9 and 3.

