**ITCS 1212L**

**Pre-Lab 4**

**Introduction to Functions and Modular Programs**

**Learning Objectives:**

* **To learn the process of creating functions and function prototypes.**
* **To learn the process of calling a function.**
* **To learn how to create a modular program.**
* **Practice developing the main algorithm which is composed of different functions calls.**

**Answer these questions:**

1. Fill-in-the-Blank Questions:
2. The word **void** precedes the name of every function proto- type and heading that does not return a value back to the calling routine.
3. Pass by **value** indicates that a copy of the actual parameter is placed in the memory location of its corresponding formal parameter.
4. **Call-by-reference** parameters are found in the call to a function.
5. A prototype must give the of its formal parameters and may give their **I’m unsure what is being asked. A prototype must declare what type of information is being sent back (if any) and what type of information is being sent to the function in question (this could be multiple values and multiple value types).**
6. A **\* (asterix)** after a data type in the function heading and in the prototype indicates that the parameter will be passed by reference.
7. Pass by **reference** indicates that the location of an actual parameter, rather than just a copy of its value, is passed to the called function.
8. A call must have the **address** of its actual parameters and must NOT have the **values** of those parameters.
9. **formal** parameters are found in the function heading.
10. What are the main components of a function? **Return Type, Function Name, Parameters, and a Function Body.**
11. When we talk about modular programming, what does that mean? **In Ley-mans terms: It’s about designing programs that have many different parts (or functions). Each function performing it’s own specific task. This way a function can be called to perform its task whenever it’s needed by the main function, and it keeps main from being clogged up with a lot of code.**
12. a. When you want to write prototype of a function, how do you start the development of it? For example, we want to create a function called makeFloat( ) that gets an integer number and returns a floating point number of the integer. For example, makeFloat(3) will return 3.0 . **The return value (float), a proper name (makeFloat), and the value being passed (int).**

b. In order to create the prototype, what do we need to specify? Create the function prototype. **Float makeFloat (int);**

c. Next, you need to create the function. How would you create the actual function? **Float makeFloat (int integerVariableBeingPassed)**

**{**

**Float floatingNumber;**

**FloatingNumber = integerVariableBeingPassed;**

**Return floatingNumber;**

**}**

d. Now, we want to write a program that tests makeFloat( ). What are the steps that we need to take to write the main program? Can you create the program and test it in the lab? **returnedFloat = floatingNumber(myIntegerFromMain);**

**cout << returnedFloat;**

1. This program prints the message you send to it. There is a function called displayMsg(string message) which gets a string as the input parameter and shows it.

#include <iostream>

using namespace std;

**//This is the function prototype for displayMsg()**

void displayMsg(string);

int main()

{

string msg;

cout << “What is your message? “;

cin >> msg;

**// Get the String and call the function**

displayMsg(msg);

return 0;

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// displayMsg() function

//

// task: This function prints a message

// data in: string message

// data out: no actual parameter altered and sent back

//

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**// Function heading**

void displayMsg(string message)

**// body of the function**

{

cout << message << endl;

}

**So, I’m not really seeing a question or a task with #5. I went over the code, and it should function as described above. I did not notice any bugs at a cursory glance. Great example for lab tomorrow.**

1. This program gets the price and the quantity and returns the cost. Fill in the code (places in Green) so that the program will print out the final cost.

#include <iostream>

using namespace std;

void calculateCost(float, int); **// prototype** (I removed p and q, as they would give a syntax error and are not needed here)

int main()

{

calculateCost(float itemCost, int quantity); // (I wasn’t asked to declare variables or retrieve info, so I’m guessing that part goes without saying)

**// Fill in the code to call the calculateCost() function**

return 0;

}

// /**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**// calculateCost()**

**//**

**// Task: gets the price and the quantity and returns the cost**

**// data in: float itemCost, int quantity**

**//**

**// data out: (Technically no data out, though the function will print the calculated answer)**

**//**

**//** /**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**// Fill in the function heading and the body of the function that // will print the total cost.**

**Void calculateCost(float itemPrice, int numberOfStuffs)**

**{**

**Float total (0);**

**total = itemPrice \* numberOfStuffs;**

**Cout << total;**

**Return 0;**

**}**