**Lab Experience Nine**

**Lab Exercises**

**Directions:**

Start Microsoft word and record the questions and answers to all of the exercises in the lab word document   
Answer the following questions based on material presented in lecture, material above and found in chapters 1-6 of the textbook.

**Exercise 1:**

Download the program **retail.cpp** from D2L. Use the program to answer the following questions.

1. Change the datatype in the statement double wholesaleCost, pctMarkup; to int and execute your program. Explain what happens after you compile and execute the program. Afterwards change it back to type double.

**Attempting to compile the program with these changes results in an error, as wholesaleCost’s formal parameters are references to doubles. If pass by value was used the ints would be automatically promoted, but the reference does not allow this.**

1. Change the datatype of the variable profit from double to an int. Explain what happens after you compile and execute the program. Afterward change it back to type double.

**Although this doesn’t give any compile errors, this does result in a logic error. As an int is being used, any calculations assigned to it are truncated before assignment. Although the function displayProfitMargin technically takes a double, it automatically promotes profit to a double before passing, unlike pass by reference.**

1. Change the statement displayRetail(retailPrice); to displayRetail(5); . Explain what happens after you compile and execute the program. Do not modify the statement until directed to do so in another question.

**Although the calculations remain correct, passing a constant literal causes the program to always display $5 for the retail price, giving a logic error.**

1. Change the prototype statement void displayRetail(double); to void displayRetail(double &); and also change the function header void displayRetail(double retailPrice) to void displayRetail(double &retailPrice) . Explain what happens after you compile and execute the program.

**As we are still passing an int literal to displayRetail, it is trying to convert from a normal int to a double reference, which it is unable to do. As it cannot convert the type, it results in a compile time error.**

1. Change the function call displayRetail(5) to displayRetail(retailPrice). Explain what happens after you compile and execute the program.

**As the function displayRetail only needs to display a value, keeping the pass by reference here doesn’t change anything, resulting in no errors of any type. If displayRetail also modified its parameter in some way then it could possibly result in a logic error, but as displayRetail only displays, the change to pass by reference here is irrelevant.**

1. Change the function call displayRetail(retailPrice) to displayRetail(retailPrice + 5). Explain what happens after you compile and execute the program. Change the statement back to displayRetail(retailPrice) after you have answered this question.

**Adding 5 to the actual parameter before passing changes it from a double reference to a double value (temp value). As the function requires a double reference, this results in a compile error when the compiler attempts to convert between the 2 types.**

1. Based upon the results of questions 1-6, what are your conclusions about passing parameters by value or by reference?

**My conclusions are that pass by reference should be used any time the value doesn’t need to be changed at the same time it’s passed to the function (like in #6). Pass by value should be used when the function needs to be able to modify the value it was passed inside the function, like performing additions on it, without modifying the original parameter. Also, pass by value should be used when a function may need a literal passed to it, such as in #5.**

**Exercise 2**

1. Download the program **printVariable.cpp** from D2L and use it to answer the following questions.

1. Insert the following comments at the top of the program.

**// Computer Science 1106**

// your name **here**

For your convenience the program is listed below:

// This program illustrates global variables.

**#include <iostream>**

**using namespace std;**

**int x; // a global variable**

**void printVariable(); // prototype**

**int main(void){**

**x = 300;**

**cout << "main before printVariable: The value in the variable x is " << x << endl;**

**printVariable();**

**cout << "main after printVariable: The value in the variable x is " << x << endl;**

**return 0;**

**}**

**//**

**// This function produces a side effect**

**//**

**void printVariable(){**

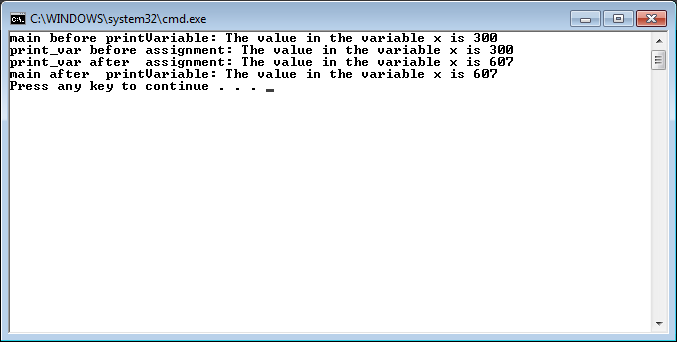
**cout << "print\_var before assignment: The value in the variable x is " << x << endl;**

**x = 2 \* x + 7;**

**cout << "print\_var after assignment: The value in the variable x is " << x << endl;**

**}**

1. Place the answers to all of the questions below in your word document.
   1. Run the program and paste the output into your word document.



* 1. Move the declaration of **x** into main. Attempt to compile the program and explain the compilation problem using complete sentences. Copy and paste the source code and output into your word document.

**Moving x’s declaration inside of main results in an undeclared identifier being thrown from printVariable because x is no longer in global scope.**

// Computer Science 1106

// Johnathan Lee

// This program illustrates global variables.

#include <iostream>

using namespace std;

void printVariable(); // prototype

int main(void) {

int x;

x = 300;

cout << "main before printVariable: The value in the variable x is " << x << endl;

printVariable();

cout << "main after printVariable: The value in the variable x is " << x << endl;

return 0;

}

//

// This function produces a side effect

//

void printVariable() {

cout << "print\_var before assignment: The value in the variable x is " << x << endl;

x = 2 \* x + 7;

cout << "print\_var after assignment: The value in the variable x is " << x << endl;

}

* 1. Fix the compilation problem by passing the variable **x** as a parameter to the function **printVariable.** Note: You also have to change the prototype and function header of **printVariable** to make this work. Why? Copy and paste the source code and output into your word document.

**I must also change the prototype and function header because otherwise technically a function called printVariable with a parameter int would not exist in the first place. Attempting to call it with only the prototype changed will result in an error because there won’t be a function definition that also has an int for a formal parameter. Attempting to call it with only the header changed will also result in an error because, at the time it was called, there would not exist a definition for printVariable that took an int parameter.**

// Computer Science 1106

// Johnathan Lee

// This program illustrates global variables.

#include <iostream>

using namespace std;

void printVariable(int var); // prototype

int main(void) {

int x;

x = 300;

cout << "main before printVariable: The value in the variable x is " << x << endl;

printVariable(x);

cout << "main after printVariable: The value in the variable x is " << x << endl;

return 0;

}

//

// This function produces a side effect

//

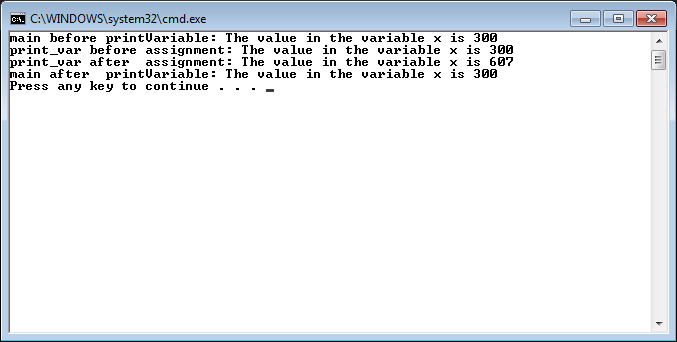
void printVariable(int var) {

cout << "print\_var before assignment: The value in the variable x is " << var << endl;

var = 2 \* var + 7;

cout << "print\_var after assignment: The value in the variable x is " << var << endl;

}



* 1. Suppose we want to return this adjusted value of **x** to main so that the output matches part a’s output (shown below). Make the necessary changes to your program. When done, paste the completed program and the output into your word document.

main before **printVariable**: The value in the variable x is 300

print\_var before assignment: The value in the variable x is 300

print\_var after assignment: The value in the variable x is 607

main after **printVariable**: The value in the variable x is 607

Press any key to continue

**(Assuming you want pass by value since you specifically said return.)**

// Computer Science 1106

// Johnathan Lee

// This program illustrates global variables.

#include <iostream>

using namespace std;

int printVariable(int var); // prototype

int main(void) {

int x;

x = 300;

cout << "main before printVariable: The value in the variable x is " << x << endl;

x = printVariable(x);

cout << "main after printVariable: The value in the variable x is " << x << endl;

return 0;

}

//

// This function produces a side effect

//

int printVariable(int var) {

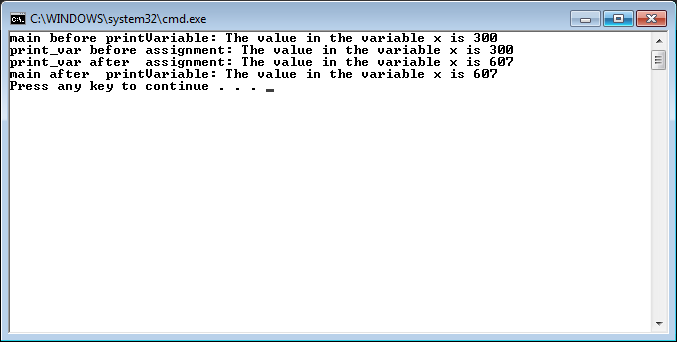
cout << "print\_var before assignment: The value in the variable x is " << var << endl;

var = 2 \* var + 7;

cout << "print\_var after assignment: The value in the variable x is " << var << endl;

return var;

}



**Exercise 3 Prime Numbers:**

A prime number is an integer greater than 1 that can only be divided evenly by itself or 1. Examples of prime numbers are: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29 ... Numbers that are not prime are: Any even number, 9 since it is divisible by 3, multiples of 5, etc.

**Note: You must use several functions in the solution of this problem ( main does not count as one of your solutions). Please note the number one is not a prime number by definition. Global variables are not allowed. main does not count as one of your functions.**  
Write a program that will accept from the user a positive integer (make sure you validate the number is positive *this would make a good function candidate.)* and determine if the number entered by the user is a prime number. Notify the user if the number is prime or it is not prime. Your program should also display the number entered by the user.   
  
**The naive or brute-force approach is to check if it is divisible by any number between 1 and itself. Use the following algorithm to determine if a number is prime or not.**

1. **If the number is even and greater than 2, it is not prime.**
2. **Moreover, you should only test the number against integers less than or equal to the square root of its own value.**

To find all kinds of interesting facts about prime numbers and the largest prime number found so far, just [click here](http://www.utm.edu/research/primes/largest.html).

Copy and paste your source code into a word document with several screen shots of all possible case scenarios. I.e. Error checking for a negative number, prime numbers of 2 and 3, and 29, a non-prime number.

// Computer Science 1106

// Programmer: Johnathan Lee

// Description: Takes an integer and says whether it's prime.

#include <cmath>

#include <iostream>

using namespace std;

void getValue(int &input);

bool isPositive(int num);

bool isEven(int num);

bool loopCheckVals(int value);

int main() {

bool prime = true; // Assume we're prime to cut down on statements.

int val;

getValue(val);

if (!isPositive(val)) {

cout << "Error, entered a negative number. Terminating.\n\n";

exit(1106);

}

if (isEven(val) && val > 2) {

prime = false;

} else {

prime = loopCheckVals(val);

}

cout << val;

if(prime)

cout << " is prime.\n";

else

cout << " is not prime.\n";

return 0;

}

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

getValue

Preconditions: An integer reference

Returns: An integer by reference.

Takes a reference to an integer, displays a prompt, then takes input

into the integer.

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

void getValue(int &input) {

cout << "Please enter a positive integer to check: ";

cin >> input;

}

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

isPositive

Preconditions: An integer

Returns: A bool

Takes an integer and returns whether or not that integer is positive (>0).

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

bool isPositive(int num) { return (num > 0); }

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

isEven

Preconditions: An integer

Returns: A bool

Uses modulo to tell whether a number is even.

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

bool isEven(int num) {

return (num % 2 == 0); // If the remainder is 0, it must be even.

}

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

loopCheckVals

Preconditions: A non-even integer

Returns: A bool

Loops from 3 to the value's sqrt to check if it's prime.

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

bool loopCheckVals(int value) {

bool isPrime = true;

// We don't need to start at 1 and we already checked for divisible by

// 2, so start at 3.

for (int i = 3; i <= sqrt(value); i++) {

if (value % i == 0)

isPrime = false;

}

return isPrime;

}

