**Lab Experience Thirteen**

**Lab Exercises**

**Directions:**

Start Microsoft word and record the questions and answers to all of the exercises in the word document   
Answer the following questions based on material presented in lecture and found in chapters 1-7 of the textbook.

**Exercise 1**

1. Given the function prototype void fun1(int &, double); is the following function call valid or invalid? Why or why not?

double x, y;

fun1(x, y);

**It is invalid because the call is attempting to pass a double (x) where the function expects a reference to an int. As the compiler cannot convert between the 2 types, a compiler error will be thrown.**

1. Given the function prototype void fun1(int &, double); is the following function call valid or invalid? Why or why not?

double x, y;

fun1(static\_cast<int>(x), 3);

**It will be invalid because the static\_cast only creates a temporary variable that will not exist after being passed along. Since the & points to a memory location, the it would either be pointing at nothing at all or be pointing at something which is non-initialized/not in the program’s control. Should be a compiler error again.**

1. Given the function prototype void fun1(int &, double); is the following function call valid or invalid? Why or why not?

int x, y;

fun1(x, y);

**This one should be valid as x is now an int. Even though y is declared as in int too, it can be cast to a double, unlike an int reference.**

1. Given the function prototype void fun1(int &, double); is the following function call valid or invalid? Why or why not?

int x, y;

double z;

fun1(x + y, z);

**This one should be invalid for the same reason as #2. x+y creates a temporary value to be passed along, but the function prototype is expecting a reference to a memory location.**

1. Given the function prototype double fun2(int , double); is the following function call valid or invalid? Why or why not?

double x, y;

fun2(x, fun2(x, y));

**This one should be perfectly valid again. In terms of types, it will look like fun2(double, double(double, double)). None of these parameters are references, so all can undergo type conversion as needed.**

**Exercise 2 Function Stubbing:**

**Description of the problem.**

The verbal part of the SAT test counts 800 points. Students can take the test several times, and colleges evaluate a prospective student based on the largest score for each part of the test. The program that you will complete will read two SAT verbal scores from the user and display the larger of the two scores.

An example of the output is listed below:

**This program interactively reads two SAT**

**verbal scores and prints the larger**

**Enter an SAT verbal score between 0 and 800 ======> 550**

**Enter an SAT verbal score between 0 and 800 ======> 600**

**The larger score is 600**

**Press any key to continue**

What you must do:

1. Download the file **functionStubbing.cpp** from D2L.
2. Start Visual C++ and load the program **functionStubbing.cpp** into your workspace. Note: If you try to compile the program, you will have two errors concerning the functions **max** and **outOfRange**.
3. Change the comments in the program using your name.
4. Follow these directions explicitly:
5. Replace the comment in the body of **readScore** with statements to convert it to a stub. A program Stub consists of an output statement stating that the function was entered correctly and if the function should return a value or change the parameter. To complete the stub place the following C++ statements in the function definition of **readScore**:

**cout << “Entering function readScore “ << endl;**

**score = 500;**

**cout << “Leaving function readScore “ << endl;**

1. Replace the comment in the body of **outOfRange** with statements to convert it to a stub.

Have the function body print something similar to part a.

**outOfRange should return false. So add the statement:** return false; **after all of your cout statements.**

1. Replace the comment in the body max of with statements to convert it to a stub. Have the function body print something similar to part a.

**max should return the value contained in the first parameter(which in our case is the variable x).**

1. Build and run your program saving it to the **file lab8.cpp**. The output should match what is listed below exactly. Paste the current source code and the output produced into your word document.

**This program interactively reads two SAT**

**verbal scores and prints the larger**

## Entering function readScore

**Leaving function readScore**

**Entering function outOfRange**

**Leaving function outOfRange**

**Entering function readScore**

**Leaving function readScore**

**Entering function outOfRange**

**Leaving function outOfRange**

**Entering function max**

**Leaving function max**

**The larger score is 500**

**Press any key to continue**

//

// Programmer: Tim Wrenn, modified by Johnathan Lee

// This was how you wanted us to do this part,

// right?

//

// File: FunctionStubbing.cpp Class: Computer Science 1106

//

// Program Description: This program reads two SAT verbal scores(between 0 and 800)

// for a student and prints the larger of the two scores.

//

#include <iostream>

using namespace std;

// Global Constants

const int MAX\_SCORE = 800;

const int LOW\_SCORE = 0;

// Function Prototypes

void instructions(void); // Function to display instructions to the user

void readScore(int &score); // Function get the scores from the user

bool outOfRange(int score); // Boolean valued function will return true or false

int max(int x, int y); // Returns the score that is the largest

int main(void){

int verbalScore1, // SAT verbal score #1

verbalScore2; // SAT verbal score #2

instructions(); // Display instructions to the user

readScore(verbalScore1); // Get the first test score

if(outOfRange(verbalScore1))

cout << verbalScore1 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else{

readScore(verbalScore2); // Get the second test score

if(outOfRange(verbalScore2))

cout << verbalScore2 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else

cout << "The larger score is " << max(verbalScore1, verbalScore2) << endl;

}

return 0;

}

//

// Function to print instructions on what the program does

//

void instructions(void){

cout << "This program interactively reads two SAT\n";

cout << "verbal scores and prints the larger\n\n";

}

//

// Function to read a score interactively

//

void readScore(int &score){

cout << "Entering function readScore " << endl;

score = 500;

cout << "Leaving function readScore " << endl;

}

//

// Function to return true if the SAT score is out of

// range, false otherwise. Range: 0-800

//

bool outOfRange(int score){

cout << "Entering function outOfRange " << endl;

cout << "Leaving function outOfRange" << endl;

return false;

}

//

// Function to return maximum of two integer parameters

//

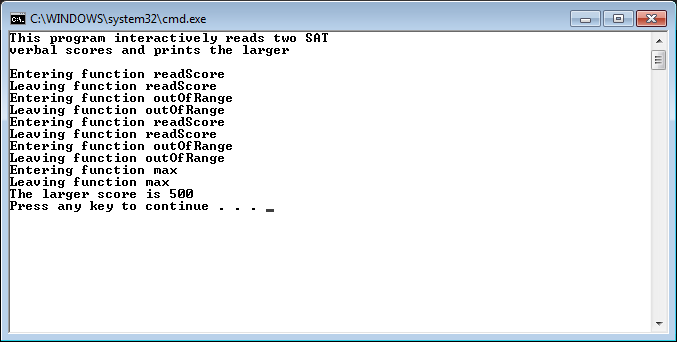
int max(int x, int y) {

cout << "Entering function max " << endl;

cout << "Leaving function max" << endl;

return x;

}



1. At this point we have tested the communication between all of our functions. Now we need to test the if statements to make sure there isn’t a logic error.
2. Change the 500 to 900 in the function **readScore** and the word false to true in the function **outOfRange**. Execute the program and your output should match the output below exactly. Paste the source code in your word document and the output into your word document.

**This program interactively reads two SAT**

**verbal scores and prints the larger**

**Entering function readScore**

**Leaving function readScore**

**Entering function outOfRange**

**Leaving function outOfRange**

**900 is not between 0 and 800.**

**Program Terminating**

**Press any key to continue**

//

// Programmer: Tim Wrenn, modified by Johnathan Lee

// This was how you wanted us to do this part,

// right?

//

// File: FunctionStubbing.cpp Class: Computer Science 1106

//

// Program Description: This program reads two SAT verbal scores(between 0 and 800)

// for a student and prints the larger of the two scores.

//

#include <iostream>

using namespace std;

// Global Constants

const int MAX\_SCORE = 800;

const int LOW\_SCORE = 0;

// Function Prototypes

void instructions(void); // Function to display instructions to the user

void readScore(int &score); // Function get the scores from the user

bool outOfRange(int score); // Boolean valued function will return true or false

int max(int x, int y); // Returns the score that is the largest

int main(void){

int verbalScore1, // SAT verbal score #1

verbalScore2; // SAT verbal score #2

instructions(); // Display instructions to the user

readScore(verbalScore1); // Get the first test score

if(outOfRange(verbalScore1))

cout << verbalScore1 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else{

readScore(verbalScore2); // Get the second test score

if(outOfRange(verbalScore2))

cout << verbalScore2 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else

cout << "The larger score is " << max(verbalScore1, verbalScore2) << endl;

}

return 0;

}

//

// Function to print instructions on what the program does

//

void instructions(void){

cout << "This program interactively reads two SAT\n";

cout << "verbal scores and prints the larger\n\n";

}

//

// Function to read a score interactively

//

void readScore(int &score){

cout << "Entering function readScore " << endl;

score = 900;

cout << "Leaving function readScore " << endl;

}

//

// Function to return true if the SAT score is out of

// range, false otherwise. Range: 0-800

//

bool outOfRange(int score){

cout << "Entering function outOfRange " << endl;

cout << "Leaving function outOfRange " << endl;

return true;

}

//

// Function to return maximum of two integer parameters

//

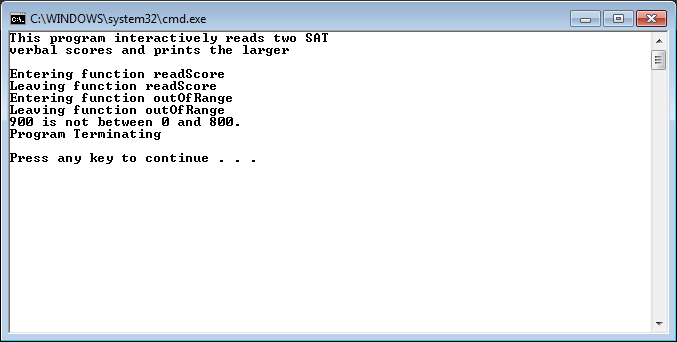
int max(int x, int y) {

cout << "Entering function max " << endl;

cout << "Leaving function max" << endl;

return x;

}



1. At this point in the Software development cycle the testing of the if statements and the functions are complete. Everything is working, so now is the time to replace the stub code with actual code to solve the problem.
2. Change the function **readScore** to appear exactly as below, which means deleting all of the stub statements that we used to test communication between the functions.

void readScore(int &score){

cout << "Enter an SAT verbal score between " << LOW\_SCORE << " and "

<< MAX\_SCORE << " ======> ";

cin >> score;

}

Since we are using call-by-reference, the actual parameters value will be changed whenever this function is called.

1. The program still doesn’t work, so we have to change the functions **outOfRange** and **max** accordingly. Change the functions so that they appear exactly as below:

bool outOfRange(int score){

return (score < LOW\_SCORE || score > MAX\_SCORE);

}

//

// Function to return maximum of two integer parameters

//

int max(int x, int y){

return (x > y ? x : y);

}

After you have completed all of the above parts, run your program using the test values below:

|  |  |  |
| --- | --- | --- |
| **Verbal SAT score number 1** | **Verbal SAT score number 2** | **Expected Results** |
| **550** | **600** | **600 is the larger** |
| **900** |  | Program terminating |
| **500** | **500** | **500 is the larger** |
| **500** | **835** | **Error Program terminating** |

**Paste the program functionStubbing.cpp into your word document. Paste the output of each run of your program using the test data above.**

//

// Programmer: Tim Wrenn, modified by Johnathan Lee

// This was how you wanted us to do this part,

// right?

//

// File: FunctionStubbing.cpp Class: Computer Science 1106

//

// Program Description: This program reads two SAT verbal scores(between 0 and 800)

// for a student and prints the larger of the two scores.

//

#include <iostream>

using namespace std;

// Global Constants

const int MAX\_SCORE = 800;

const int LOW\_SCORE = 0;

// Function Prototypes

void instructions(void); // Function to display instructions to the user

void readScore(int &score); // Function get the scores from the user

bool outOfRange(int score); // Boolean valued function will return true or false

int max(int x, int y); // Returns the score that is the largest

int main(void){

int verbalScore1, // SAT verbal score #1

verbalScore2; // SAT verbal score #2

instructions(); // Display instructions to the user

readScore(verbalScore1); // Get the first test score

if(outOfRange(verbalScore1))

cout << verbalScore1 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else{

readScore(verbalScore2); // Get the second test score

if(outOfRange(verbalScore2))

cout << verbalScore2 << " is not between " << LOW\_SCORE << " and "

<< MAX\_SCORE << ". \nProgram Terminating" << endl << endl;

else

cout << "The larger score is " << max(verbalScore1, verbalScore2) << endl;

}

return 0;

}

//

// Function to print instructions on what the program does

//

void instructions(void){

cout << "This program interactively reads two SAT\n";

cout << "verbal scores and prints the larger\n\n";

}

//

// Function to read a score interactively

//

void readScore(int &score){

cout << "Enter an SAT verbal score between " << LOW\_SCORE << " and "

<< MAX\_SCORE << " ======> ";

cin >> score;

}

//

// Function to return true if the SAT score is out of

// range, false otherwise. Range: 0-800

//

bool outOfRange(int score){

return (score < LOW\_SCORE || score > MAX\_SCORE);

}

//

// Function to return maximum of two integer parameters

//

int max(int x, int y) {

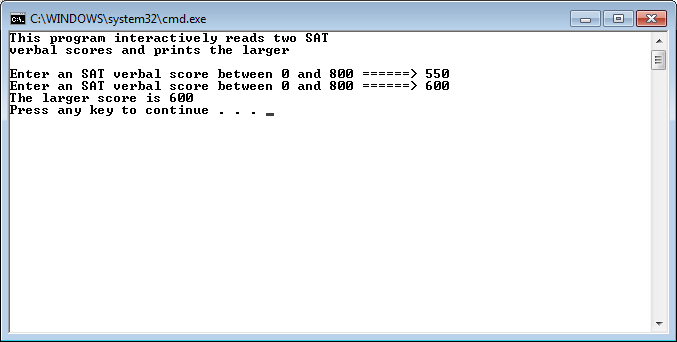
return (x > y ? x : y);

}

Run #1:

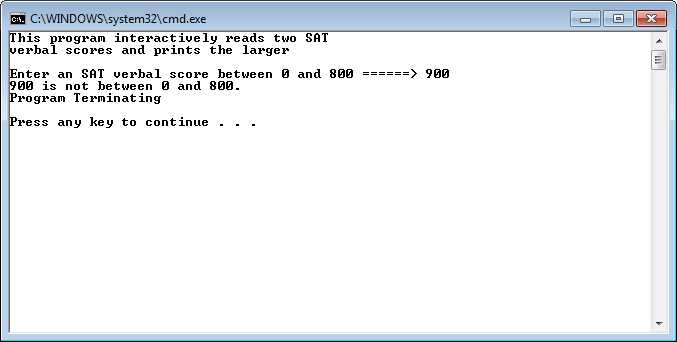
Test Data: 500, 600

Results:



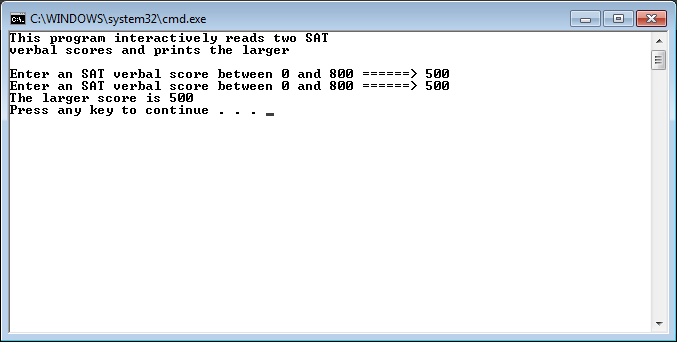
Run #2:

Test Data: 900, (none)



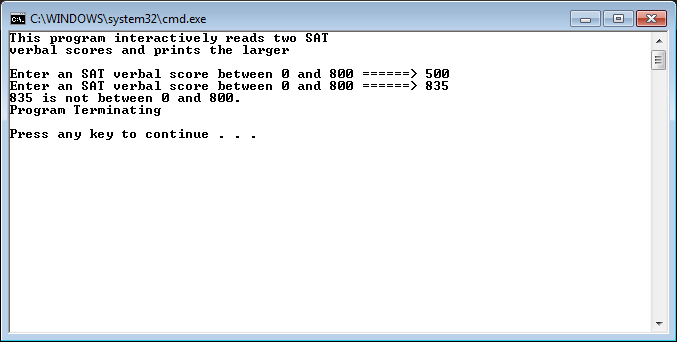
Run #3:

Test Data: 500, 500



Run #4:

Test Data: 500,835



**Exercise 3: Common errors involved in passing arrays to functions.**

Download the following program, Array from D2L and load it into your compiler.

// This program demonstrates an array being passed to a function.

#include <iostream>

using namespace std;

void showValues(int [], int); // Function prototype

const int ARRAY\_SIZE = 100;

int main()

{

int numbers[ARRAY\_SIZE] = {5, 10, 15, 20, 25, 30, 35, 40};

showValues(numbers, 8);

return 0;

}

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

// Definition of function showValue. \*

// This function accepts an array of integers and \*

// the array's size as its arguments. The contents \*

// of the array are displayed. \*

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

void showValues(int nums[], int size)

{

for (int index = 0; index < size; index++)

cout << nums[index] << " ";

cout << endl;

}

Perform the following operations and record your results (error messages, etc.) in your word document. **After you have recorded your results from the error message restore the program back to its original code.**

1. Change void showValues(int [], int); to void showValues(int , int); and execute your program.

**Error: void showValues(int, int): cannot convert from int[100] to int.**

**Here the compiler expected a single int, but instead got an array, causing a compiler error.**

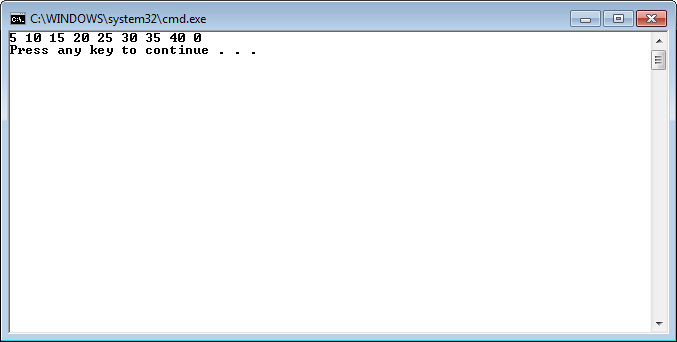
1. Change void showValues(int nums[], int size) to void showValues(int nums, int size) and execute your program.

**Subscript requires array or pointer type. Basically the opposite of the above problem. For some reason the compiler didn’t complain about the lack of [] in the function definition, instead throwing an error when the program tries to use subscript notation on a single variable.**

1. Change for (int index = 0; index < size; index++) to

for (int index = 0; index <= size; index++) and execute your program.

**Outputs:**



**Here, attempting to grab an uninitialized value got a 0. No compiler errors, just a logic error.**

1. Change showValues(numbers, 8); to showValues(numbers[], 8); and execute your program.

**Syntax error: ].**

**I’m guessing that it expected there to be something before the ‘]’ to grab another value that would be passed to the function instead, but when it found the ‘]’ early it threw an error.**

1. Change showValues(numbers, 8); to showValues(numbers[0], 8); and execute your program.

**Cannot convert from type int to int[].**

**Here, the function expected an array but instead got a single variable. Kind of like the opposite of #1.**

1. Change void showValues(int [], int); to void showValues(int &, int); and change

void showValues(int nums[], int size) to void showValues(int &nums, int size) and execute your program.

**Cannot convert from int[] to int& and subscript requires an array or pointer.**

**Here, the compiler is unable to convert from an int array to an int reference, and so throws an error. Also since references are treated like normal variables in terms of syntax, trying to use subscript notation on an int reference caused an error.**

1. Change showValues(numbers, 8); to showValues(numbers, ARRAY\_SIZE); and execute your program.

**From the original program, it can be inferred that we only want to print out from 5 to 40, so printing all 100 variables (ARRAY\_SIZE) should be a logic error.**

**Due Date:**

As indicated on the Lab Ten Dropbox Folder.

**What to hand in:**

1. Save your word document as yournamelab10.docx, i.e. timwrennlab10.docx.
2. Compress your program from .cpp file and your word file into a single compressed folder called yournamelab10.zip.
3. Hand in printouts of your program and your word document.
4. Place the compressed file into the lab 10 assignment folder.