**Name:**

**Advanced Programming in C++**

**Lab Exercise 3/24/2020**

In this exercise you will use pointers and references to solve a variety of problems. For each of these exercises, you are to print out your source code as well as a sample of your output.

1. Write a program that creates an array of 100 integers. Pass the array to a function

void printArray(int \*);

The function should use pointer notation to print the array.

Hint: Consider using a pointer placed at the end of the array as such:

pLast = arrayName + SIZE – 1 //places pointer at last array element

1. Write a program that allows the user to enter two integers and passes them to three swap functions. User the following three prototypes for your swap functions:

void swap1(int, int); //pass by value

void swap2(int \*, int \*); //pass a pointer

void swap3(int &, int &); //pass a reference

Write a main function that calls each of these functions printing out the two values prior to and after the call to each.

1. Write a program that will do the following:

* Create an array of 100 elements
* Initialize the array with random values from 1 to 1000
* Print the array elements to the screen using pointers
* Print the array elements to the screen in reverse order using pointers
* Find the largest value of the array using pointers
* Find the smallest value of the array using pointers
* Find the average value of the array using pointers

To get you started, here are the prototypes for the functions as well as a main function:

//Prototypes

void initialize(int \*);

void display(int \*);

void displayBackwards(int \*);

int maximum(int \*);

int minimum(int \*);

double average(int \*);

//Main Function

int main()

{

int numbers[SIZE];

srand(time(NULL));

initialize(numbers);

display(numbers);

cout << endl << "The array backwards" << endl;

displayBackwards(numbers);

cout << endl;

cout << "The largest value is: " << maximum(numbers) << endl;

cout << "The smallest value is: " << minimum(numbers) << endl;

cout << "The average value is: " << average(numbers) << endl;

return 0;

}