CMPT 125 Week 5 Lab Work

1. Consider the following program

```
int main()
{
    //Read the coefficients of the quadratic equation ax^2 + bx + c = 0
    //Assume the user input value for the coefficient a is different from zero
    double a, b, c;
    cout << "Enter the coefficients a, b, c: ";
    cin >> a >> b >> c;

    //Prepare two memory spaces where to put the solutions of the quadratic equation, if any
    double s1, s2;

    //Call a function that will compute the real solutions of the quadratic equation, if any,
    //and then stores the computed real solutions in the memory spaces named s1 and s2
    //Design the function such that it also returns the number of real solutions
    int n = quadraticSolver(a, b, c, &s1, &s2);
    if (n==0)
           cout << "The quadratic equation has no real solution" << endl;
    else if (n==1)
    {
           cout << "The quadratic equation has one real solution " << s1 << endl;
           cout << "The second same solution is  " << s2 << endl;
    }
    else
           cout << "The quadratic equation has two real solutions " << s1 << " and " << s2 << endl;

    system("Pause");
    return 0;
}
```

   Implement the **quadraticSolver** function that takes the coefficients of the quadratic equation and the memory addresses of two variables that will store the real solutions of the quadratic equation (if any) computed inside the function and that returns how many real solutions the quadratic equation has.

2. The previous function uses parameter passing by pointer for both the solutions. Modify the program and your function so that this time to use parameter passing by reference for both the solutions.

3. The previous function uses parameter passing by pointer for both the solutions. Modify the program and your function so that this time to use parameter passing by reference for one the solution and parameter passing by reference for the other solution.

4. Write a program that creates a C++ dynamic array of integers of user desired size, populates the elements of the array with random integers in the range [-10, 10] and then prints the maximum and minimum elements of the array. Please note that the array size must be read from the user as follows:

```
int arraySize;
cout << "Please enter the size of the array to be created: ";
cin >> arraySize;
```

5. Define a function that takes a C++ dynamic array of integers and its size as arguments and returns the number of prime numbers in the array. Assume all the elements of the array are positive integers greater than 1.

   Test your function by writing a main program that creates a dynamic array of integers of user defined size, populates the array with random integers in the range [2, 100] and calls the function to determine how many prime number elements are there in the array and finally prints the message "A prime number is

found the array " if there is at least one prime number element in the array; otherwise it prints the message "No prime number is found the array ".

You should define a helper function **`bool isPrime(int a)`** function to answer this question easily.

6. Define a function that takes a C++ dynamic array of characters and its size as arguments and returns the number of alphabets in the array. Alphabet means any upper case or lower case English letter.

   Test your function by writing a main program that creates a dynamic array of characters of user defined size, populates the array with random characters whose ascii codes are in the range [48, 122] and calls the function to determine how many alphabet character elements are there in the array and finally prints the message "An alphabet is found the array " if there is at least one alphabet character element in the array; otherwise it prints the message "No alphabet character is found the array".

7. Define a function that takes an array of integers and its size as arguments and returns true if all the elements of the array are equal and returns false otherwise.

8. Define a function that takes an array of integers and its size as arguments and returns true if all the elements of the array are different from each other and returns false otherwise.

9. Define a function that takes a dynamic array of floats and its size and returns a new array of floats whose elements are the elements of the argument array in reverse order.

10. Define a function that takes two dynamic arrays of integers **A** and **B** and their sizes **`sizeA`** and **`sizeB`** respectively and returns a new dynamic array of integers of size **`sizeA+sizeB`** and whose elements are those elements of A followed by the elements of B.

11. Define a function named **`isFound`** that takes a dynamic array of integers, its size and an integer value as arguments and that returns true if the integer number is found in the array otherwise it returns false.

12. Define a function named **`countElements`** that takes two dynamic arrays A and B of integers and their sizes as arguments and returns the number of elements A that are found in B. That is this function returns how many elements of A are found in B. Use your **`isFound`** function defined above.

13. Define a function named **`isContained`** that takes two dynamic arrays A and B of integers and their sizes as arguments and returns true if every element of A is found in B; otherwise returns false. Use your **`countElements`** function defined above.

14. Define a function that takes two dynamic arrays A and B of integers and their sizes as arguments and returns true if every element of A is found in B and that every element of B is found in A; otherwise returns false. Use your **`isContained`** function defined above.

15. Define a function that takes two dynamic arrays of integers A and B and their sizes as arguments and returns a new dynamic array of integers whose elements are the elements of A that are found in B. Note that this function must return a dynamic array.

    In your test main program, you will notice that the main program will not know how many elements the returned dynamic array has. This is because we only returned the dynamic array from the function but not its size. Moreover we can't return both the dynamic array and its size because a function can return only one value. So what should we do? Well we should declare the size of the array in the main program and pass it by reference to the function and then the function must assign it the size of the dynamic array created in the function. In conclusion we see that the function must take FIVE arguments; namely array A, its size sizeA, array B, its size sizeB, and integer argument size (by reference). Then this function must

    - Count how many elements of A are found B.

- Assign the count to the parameter size.
- Create a new dynamic array of integers of the size just computed.
- Fill the array with elements of A that are found in B.
- Return the dynamic array.

16. Analyze the following complete C++ program carefully and determine what the output of the program is. You should not type the code in a computer for otherwise you won't learn anything. First carefully analyze what each function does and then analyze the main program.

```
#include <iostream>
using namespace std;

void printArray(const int *arr, const int arr_size)
{
	for (int i = 0; i < arr_size; i++)
		cout << "Element at " << i << " = " << arr[i] << endl;
}
void deleteElement(int* &arr, int &arr_size, const int index)
{
	if (index < 0 || index >= arr_size)
		return;
	else
	{
		//Create a new dynamic array of one less size on the heap memory
		int *B = new int[arr_size-1];

		//Copy elements of the given array except the element to be deleted to the new dynamic array
		for (int i = 0; i < index; i++)
			B[i] = arr[i];
		for (int i = index+1; i < arr_size; i++)
			B[i-1] = arr[i];

		//Delete the given dynamic array from the heap memory
		delete [] arr;

		//Assign the new dynamic array to the given dynamic array
		arr = B;

		//Decrement the size of the given dynamic array by 1
		arr_size--;
	}
}
int main()
{
	//Create a dynamic array and print it
	int size = 8;
	int *A = new int[size];
	for (int i = 0; i < size; i++)
		A[i] = 15+i;
	cout << "Originally the array is..." << endl;
	printArray(A, size);

	//Call deleteElement function to delete an element and then print the array
	int index = -5;
	deleteElement(A, size, index);
	cout << "After calling the deleteElement function with index = -5, the array is..." << endl;
	printArray(A, size);

	//Call deleteElement function to delete an element and then print the array
	index = 3;
	deleteElement(A, size, index);
	cout << "After calling the deleteElement function again with index = 3, the array is..." << endl;
	printArray(A, size);

	//Delete the dynamic array
	delete [] A;

	system("pause");
	return 0;
}
```

17. One problem with dynamic arrays is that once the array is created using the new operator then we can't shrink or expand the allocated memory. But what if we want to append (that is to insert one more element at the end of the array) which needs the allocated memory to be expanded?

Define a function named **appendElement** that takes a dynamic array of integers, its size and an integer value as arguments and that appends the integer argument to the dynamic array. For the sake of simplicity the function declaration is provided below.

```
void appendElement(int* &arr, int &arr_size, const int e)
```

Now, use the following program to test your function.

```
int main()
{
	int size = 0;
	int *A;
	for (int i = 0; i < 5; i++)
	{
		appendElement(A, size, 15+i);
		for (int i = 0; i < size; i++)
			cout << A[i] << "  ";
		cout << endl;
	}
	delete [] A;

	system("pause");
	return 0;
}
```

Required Output

```
15
15  16
15  16  17
15  16  17  18
15  16  17  18  19
```

EXTRAS

1. **[Tricks of C++ Language]** What is the difference between a dynamic array variable and a static array variable?

a. What is the data type of a dynamic array variable?

Ans:- A pointer.

b. Whose memory address does a dynamic array variable store?

Ans:- The memory address of the first element.

c. Is the memory address of the dynamic array variable different from your answer in part (b)?

Ans:- YES, it is different!!!

d. What is the data type of a static array variable?

Ans:- A pointer.

e. Whose memory address does a static array variable store?

Ans:- The memory address of the first element.

f. Is the memory address of the static array variable different from your answer in part (e)?

Ans:- NO, it is not different!!! This is advanced topic. No need to worry if you don't get it.

In order to test the above explanation, copy and paste the following program and run it to convince yourself of what is going on. Moreover draw memory diagrams to understand this concept better.

```
int main()
{
	int *A = new int[3];	//Dynamic Array
	int B[3];		//Static Array

	cout << A << endl;	//Value of dynamic array
	cout << &(A[0]) << endl;	//memory address of first element
	cout << &A << endl;	//memory address of dynamic array

	cout << endl;

	cout << B << endl;	//Value of static array
	cout << &(B[0]) << endl;	//memory address of first element
	cout << &B << endl;	//memory address of static array

	delete [] A;
	system("Pause");
	return 0;
}
```

2. What is the output of the following program?

```
#include <iostream>
using namespace std;
void figure_me_out(int& x, int y, int& z)
{
	cout << "Entering function: " << x << ", " << y << ", " << z << endl;
	x = 1;
	y = 2;
	z = 3;
	cout << "Exiting function: " << x << ", " << y << ", " << z << endl;
}
int main()
{
	int a = 10, b = 20, c = 30;
	cout << "Main Program before function call: " << a << ", " << b << ", " << c << endl;
	figure_me_out(a, b, c);
	cout << "Main Program after function call: " << a << ", " << b << ", " << c << endl;
	system("Pause");
	return 0;
}
```

3. What is the output of the following code fragment when embedded in a valid C++ main program?

```
int a = 1, b = 2, c =3;
int *d = &a, *e = &b, *f = &c;
int &g = a;
int &h = b;
int &k = c;
*e = k;
e = &h;
*d = h;
cout << g << ", " << b << ", "  << *f << endl;
```

4. What is the output of the following program?

```
void magic(int &a, int *b, int c)
{
	c = a;
	*b = a + c;
	a = c * *b;
```

```
}
int main()
{
	int x = 6, y = 8, z = 10;
	cout << x << "   " << y << "   " << z << endl;
	magic(y, &x, z);
	cout << x << "   " << y << "   " << z << endl;
	system("Pause");
	return 0;
}
```

5. Let's repeat Q#4 above but now with different variable names. Is there any syntax error in this program? What is the output of the following program?

```
void magic(int &a, int *b, int c)
{
	c = a;
	*b = a + c;
	a = c * *b;
}
int main()
{
	int a = 6, b = 8, c = 10;
	cout << a << "   " << b << "   " << c << endl;
	magic(b, &a, c);
	cout << a << "   " << b << "   " << c << endl;
	system("Pause");
	return 0;
}
```

6. Consider the following program and determine its output

```
void foo(int &m, int n, int *x, int *y)
{
	x = y;
	*y = 7;
	m = 8;
	n = *x;
	cout << m << ", " << n << ", " << *x << ", " << *y << endl;
	system("Pause");
	return;
}
int main()
{
	int a = 3, b = 5;
	int *p = &a, *q = &b;
	cout << *p << ", " << *q << endl;
	foo(a, b, p, q);
	cout << a << ", " << b << endl;
	system("Pause");
	return 0;
}
```

7. What is the output of the following C++ program?

```
int main()
{
	int a = 6, b = 8;
	int *p = &a, *q = &b, **r = &p;
	*p = *q;
```

```
        **r = 10;
        cout << a << ", " << b << ", " << *p << ", " << *q << ", " << **r << endl;
        *q = 12;
        *p = **r;
        cout << a << ", " << b << ", " << *p << ", " << *q << ", " << **r << endl;
        system("Pause");
        return 0;
}
```

8. Consider the following program and determine its output. Also identify the line of code that must be removed from the program to avoid run time error.

```
int main()
{
        int *a = new int(7);//assume the heap memory has address 4F
        int *p;
        p = a;
        cout << a << endl;
        cout << p << endl;
        cout << *a << endl;
        cout << *p << endl;
        *p = 10;
        cout << *a << endl;
        delete p;
        delete a;
        system("Pause");
        return 0;
}
```

9. Consider the following program. Find the lines of code that must be removed from the main function or from the magic function; and also add some lines of code to the main function or the magic function so that the program runs correctly with no run-time error and that all the memory reserved in the heap will be cleared before the program ends.

```
int* magic()
{
        int *p = new int(7); //assume the heap memory has address 4F
        cout << p << endl;
        cout << *p << endl;
        delete p;
        cout << *p << endl;
        cout << p << endl;
        return p;
}
int main()
{
        int *a;
        a = magic();
        cout << a << endl;
        cout << *a << endl;
        delete a;
        *a = 6;
        cout << *a << endl;
        system("Pause");
        return 0;
}
```

10. Consider the following program. What is its output? What does the function foo return?

```
 int& foo(int *p)
```

```
{
	*p = 7;
	return *p;
}
int main()
{
	int x = 5;
	int y = foo(&x);
	cout << x << ", " << y << endl;
	x = 12;
	cout << x << ", " << y << endl;
	y = 15;
	cout << x << ", " << y << endl;

	system("Pause");
	return 0;
}
```

11. What is the output of the following program?

```
int& magic(int &a, int *b, int c)
{
	c = a;
	*b = a + c;
	a = c * *b;
	return a;
}
int main()
{
	int x = 6, y = 8, z = 10;
	cout << x << "   " << y << "   " << z << endl;
	z = magic(y, &x, z);
	cout << x << "   " << y << "   " << z << endl;
	system("Pause");
	return 0;
}
```

12. What is the output of the following program?

```
int& foo(int *p)
{
	*p = 7;
	return *p;
}
int main()
{
	int x = 5;
	int& y = foo(&x);
	cout << x << ", " << y << endl;
	x = 12;
	cout << x << ", " << y << endl;
	y = 15;
	cout << x << ", " << y << endl;
	system("Pause");
	return 0;
}
```

13. What is the output of the following program?

```
int& foo(int *p)
{
```

```
        int* q = new int;
        *q = *p;
        return *q;
}
int main()
{
        int x = 5;
        int& y = foo(&x);
        cout << x << ", " << y << endl;
        x = 12;
        cout << x << ", " << y << endl;
        y = 15;
        cout << x << ", " << y << endl;
        delete &y;   //This will delete the heap memory created in foo function
        system("Pause");
        return 0;
}
```

14. Now consider the following program. If you type the code and run it, it may run fine. But it has a run time error. Explain the error.

```
int& foo(int *p)
{
        *p = 7;
        int q = *p;
        return q;
}
int main()
{
        int x = 5;
        int y = foo(&x);
        cout << x << ", " << y << endl;
        system("Pause");
        return 0;
}
```

15. Now consider the following program. If you type the code and run it, it may run fine. But it has a run time error. Explain the error.

```
int& foo(int *p)
{
        *p = 7;
        int q = *p;
        return q;
}
int main()
{
        int x = 5;
        int& y = foo(&x);
        cout << x << ", " << y << endl;

        system("Pause");
        return 0;
}
```

16. What is the output of the following program?

```
int& foo(int *p)
{
        *p = 7;
```

```
        return *p;
}
int main()
{
        int x = 5;
        foo(&x) = 3;
        cout << x << endl;

        system("Pause");
        return 0;
}
```

17. What is the output of the following program?

```
int& magic(int &a, int *b, int c)
{
        c = a;
        *b = a + c;
        a = c * *b;
        return a;
}
int main()
{
        int x = 6, y = 8, z = 10;
        cout << x << "   " << y << "   " << z << endl;
        magic(y, &x, z) = 5;
        cout << x << "   " << y << "   " << z << endl;
        system("Pause");
        return 0;
}
```

18. What is the output of the following program?

```
int& magic(int &a, int *b, int c)
{
        c = a;
        *b = a + c;
        a = c * *b;
        return a;
}
int main()
{
        int x = 6, y = 8, z = 10;
        cout << x << "   " << y << "   " << z << endl;
        magic(y, &x, z) = z;
        cout << x << "   " << y << "   " << z << endl;
        system("Pause");
        return 0;
}
```

19. Is there any syntax error in the following program? Recall that function overloading requires two or more functions with the same name to have different signatures. Which of the following functions have same signatures?

```
void foo(int x, int y)
{
        cout << x << "  " << y << endl;
        return;
}
void foo(int *x, int *y)
```

```
{
	cout << x << "  " << y << endl;
	return;
}
void foo(int &x, int &y)
{
	cout << x << "  " << y << endl;
	return;
}
```

20. What is the output of the following code fragment? The code is assumed to be embedded in a correct and complete program.

```
int *a = new int[10];
int *p = a;
int i;
for (i = 0; i < 10; i++)`
   a[i] = i;
for (i = 0; i < 10; i++)
cout << p[i] << " ";
cout << endl;
delete [] p;
```

21. What is the output of the following code fragment? The code is assumed to be embedded in a correct and complete program.

```
int array_size = 10;
int *a;
a = new int [array_size];
int *p = a;
int i;
for (i = 0; i < array_size; i++)
	a[i] = i;
p[0] = 10;
for (i = 0; i < array_size; i++)
	cout << a[i] << " ";
cout << endl;
delete [] a;
```