**DALHOUSIE UNIVERSITY**
**DEPARTMENT OF ENGINEERING MATHEMATICS**
**ENGM3282: DATA STRUCTURES AND NUMERICAL METHODS**

**ASSIGNMENT # 1, Due date: Tuesday, September 18, 2018, 1:00 PM**
For marking purposes, copy and paste your source code, output files and/or screen output into the file `solutions1.txt` and attach this file to your submission on Brightspace.

1. The polar coordinates $r$ and $\theta$ of a point $(x, y)$ in the plane can be computed as follows:

$$r = \sqrt{x^2 + y^2}$$

$$\theta = \begin{cases} \text{acos}(x/r) & \text{if } r > 0 \text{ and } y > 0 \\ -\text{acos}(x/r) & \text{if } r > 0 \text{ and } y \leq 0 \\ 0 & \text{if } r = 0 \end{cases}$$

Write a program which reads the coordinates (floats), $(x, y)$ from a the file `polar.txt` then calls the function `polar` to computes the polar coordinates. The program then writes the polar coordinates to the output file `polarout.txt`.

**Your function must use pass by reference not pass by address.**

For example, if `polar.txt` consists of:

```
 1  2
-1  2
-1 -2
 1 -2
```

then the output file `polarout.txt` will be:

```
2.23607 1.10715
2.23607 2.03444
2.23607 -2.03444
2.23607 -1.10715
```

Here is a template to get started:

```
/* File: polar.cpp
   This is a driver program for the function polar which computes the
   polar coordinates of a point in the plane

   Programmer:                              Date:
*/

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

```
int main(void)
{
    float x, y, r, theta;



    return 0;
}
```

2. When we want to intialize a dynamically allocated float array we may choose one of the following methods:

   - The array size is not specified so it is intialized to 1 and the values stored in the array is set to 0.0.
   - The array is allocated to a specified size but the values stored in the array are not specified. In this case, all the values in the array are set to 0.0.
   - The array is allocated to a specified size and the values stored in the array are all set to a specified value.

   Write the three versions of the `init()` function in the following program:

```
/*  File: intializearray.cpp
    This program uses a number of different functions to initialize an array

    Programmer: your name             Date:
*/

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

/* prototypes */



int main(void)
{
    float* x;
    float val;
    int n;

    n = 1;
    x = init();

    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;
```

```
    n = 4;
    x = init(n);

    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;

    n = 5;
    val = 1.27;
    x = init(n, val);


    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;

    return 0;
}
```

The output of your program will be:

```
x =
0

x =
0
0
0
0

x =
1.27
1.27
1.27
1.27
1.27
```

3. Modify the previous program to use a single function with default arguments, the output of the program should not change.

```
/*  File: intializearraydefaults.cpp
    This program uses a function with default arguments to initialize an array

    Programmer: your name             Date:
*/
```

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

/* prototype */



int main(void)
{
    float* x;
    float val;
    int n;

    n = 1;
    x = init();

    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;

    n = 4;
    x = init(n);

    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;

    n = 5;
    val = 1.27;
    x = init(n, val);


    cout << "\nx = \n";
    for(int i=0; i< n; i++) {
        cout << x[i] << endl;
    }

    delete [] x;

    return 0;
}
```

4. Complete the program `minmax.cpp` which uses two functions `readdata` and `minmax` to find the maximum and minimum of a set of floats stored in the file `minmax.txt`. You can assume that the number of values in the file is less than 100.

This first number in this file is an `int` indicating the number of floats which follow.

The function `readdata` opens the input file `minmax.txt` reads the first number, `n`, and then reads the following `n` floats into an array. The file is then closed by the function.

The function `minmax` computes the minimum and maximum values found in the array.

The main program prints the maximum and minimum. The content of the output file `minmaxout.txt` will be:

```
The array has 80 elements
The maximum value in the array is 63.606
The minimum value in the array is 4.8089
```

```
/* File: minmax.cpp

   Programmer:                                Date:
*/


#include <iostream>
#include <fstream>

using namespace std;

const int N = 100; // maximum array size

void readdata(/* you fill in here */);
void minmax(/* you fill in here */);

int main(void)
{
    ofstream outfile("minmaxout.txt");

    float x[N];             // the array
    int n ;                 // the actual array size
    float max;              // the maximum
    float min;              // the minimum

    /* read the data into the array */
    readdata(/* you fill in here */);

    /* compute the maximum and minimum */
    minmax(/* you fill in here */);

    outfile << "The array has " << n << " elements\n";
    outfile << "The maximum value in the array is " << max << endl;
    outfile << "The minimum value in the array is " << min << endl;

    outfile.close();

    return 0;
}

// put your function definitions here
```

5. Modify the previous program so that the array is dynamically allocated within the function `void readdata()`. Call your new program `minmaxd.cpp` with output file `minmaxdout.txt`. Your function `void readdata()` must not return anything. The output of the program will not change.