SEHH2042 Computer Programming

Tutorial 5 – Functions (1)

Q1. Write a program **DisplaySinCos.cpp** that displays the sine values and cosine values of degree from 0 to 360, with increments of 10 degrees. Note that between two columns there are <u>two tab intervals</u>.

```
Degree          Sin             Cos
0               0               1
10              0.173648                0.984808
20              0.34202         0.939693
30              0.5             0.866025
......
350             -0.173648               0.984808
360             -7.17959e-009           1
```

<u>Hints</u>:

(i) To calculate sine and cosine values, the following Math library functions (in <cmath>) can be used:

```
double sin(double radian);
double cos(double radian);
```

(ii) The above functions take in "radian" values as argument. To use them the degree value has to be **converted to radian** value first:

$$radian = degree \times \frac{\pi}{180}$$

You may need to declare a constant variable "**const double PI = 3.14159265"** in your program.

Think about the following first:

(1) What header files/library are required?
(2) How many variables are needed?
(3) What is/are the data type(s) of variable(s)?
(4) What are the steps and calculations involved?

Follow the steps below to write the program:

1. <u>Include necessary header files/library and write the main block.</u>

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

int main()
{
    // code of subsequent steps

    return 0;
}
```

<cmath> library for sin() and cos().

2. Declare necessary variables.

```
const double PI = 3.14159265;
double radian;
```

3. Print the heading line, which does not need to repeat by loop.

```
cout << "Degree\t\tSin\t\tCos\n";
```

Two tab intervals, i.e. "\t\t".

4. Use a for loop to go through 0 degree to 360 degree, with the increment of 10 degree.

```
for (int degree = 0; degree <= 360; degree += 10)
{
    // code to display sine and cosine values
}
```

5. In the for loop body, i.e. for each degree value, calculate the radian and display the sine and cosine of that degree.

```
radian = degree * PI / 180;
cout << degree << "\t\t" << sin(radian) << "\t\t" << cos(radian) << endl;
```

To call the sin() and cos() functions, do not need to write the data type in the prototype.

Think about this:

```
radian = degree / 180 * PI;
```

generate same result?

Q2. Due to different lengths of the display text, using tab intervals do not align the content well. Modify the program in Q1 to produce a formatted table as below. In the output, (1) degree values are right justified; (2) sine and cosine values are in FOUR decimal places; (3) positive and negative signs are added in the sine and cosine values.

```
Degree          Sin             Cos
  0             +0.0000         +1.0000
 10             +0.1736         +0.9848
 20             +0.3420         +0.9397
 30             +0.5000         +0.8660
......
350             -0.1736         +0.9848
360             -0.0000         +1.0000
```

Hints:

(i) For controlling the output format, include the <iomanip> header.

(ii) You may find the following stream manipulators useful: **setw(n)**, **setprecision(n)**, **fixed**, **right**

(iii) The stream manipulator **showpos** specifies positive floating points are preceded by a plus (+) sign, **noshowpos** specifies the plus sign is not shown.

Q3. In tutorial 4, you wrote a program that calculates the value of $base^{exponent}$. Now, write a function **integerPower()** that performs similar task. The function accepts a **floating point** arguments *base* and an **integer** argument *exponent*, and returns the value of $base^{exponent}$. **DO NOT** use any Math Library functions in your code. Write a program that produces the sample results below, to demonstrate the use of the function.

Note: You need to consider all possible cases of exponent: *positive, zero* and *negative*.

Sample result:

```
Enter the base value: 2.5
Enter the exponent value: -2
2.5 to the power -2 is 0.16
```

Q4. Consider the following user-defined function **nChar()** with 2 parameters:

(1) **n** – the number of characters to be printed, and
(2) **c** – the type of character to be printed.

```
void nChar(int n, char c)
{
      for (int i = 0; i < n; i++)
            cout << c;
}
```

Without changing the user-defined function **nChar()**, and by calling the function **nChar()** appropriately, complete the main function below so that the following patterns can be printed. The given code can be downloaded from Moodle.

Note that the patterns in (a) to (e) are printed separately.

```
int main()
{
      // Your codes should be inserted here

      return 0;
}
```

(a) Sample result:

```
*
**
***
****
*****
```

(b) Sample result:

```
    *
   **
  ***
 ****
*****
```

(c) Sample result:

```
    *
   ***
  *****
 *******
*********
```

(d) Sample result:

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

(e) Sample result:

```
    *
   ***
  *****
 *******
*********
    *
   ***
  *****
 *******
*********
    *
   ***
  *****
 *******
*********
```