IC 150 P Computation for engineers lab

Lab assignment sheet no: 5, Odd semester, 2016

Advanced Functions

Prepared by: Siddhartha Sarma, Maben Rabi

Objective for this lab session

- More practice with breaking up a computational task into smaller units, and to write a series of appropriate functions.
- Learn to use the `switch` statement.

Note:

- Your program should perform *exception handling* explicitly. (SEE: page 3 of this assignment sheet) For example, if you have to calculate $\log(x)$, then you should first check that user has supplied a x that is a number, and secondly check that it is non-negative. If the input is valid, print out the desired output. Otherwise, exit the program after printing an *appropriate* error message.
- Whenever possible, try to divide your `main()` function in the following subparts: `input()`, `compute()` and `output()`. You may use inline comments (using '//') or block comments (using the pair '/*' and '*/') to make the divisions explicit.

Task one: Polynomial approximation of functions

Part A: Find an approximate value of a given function $f(x)$ by computing its MacLaurin's series[1] expansion up to the nth term. Your program must give the user a choice of evaluating one of the functions:

$$\exp(x), \sin(x), \cos(x), \tan(x), \cot(x), \log(x)$$

You may use the built-in C function `float pow(float x, float y)`.

`float nthCoefficient (unsigned int n, unsigned int chosenFunction )` to calculate the coefficient of x^n in the MacLaurin's series expansion for the function chosen by the user. The unsigned interger `chosenFunction`

`int Fact(int n )` to calculate factorial of n

Part B: Write a program where you will use the function you have written in part A (let us name it `Myfunc(x, n)`) to compare its value with the corresponding built-in function for different values of n. For example, if you are evaluating an approximate value for *exponential* function using `Myfunc(x, n)`, then you are supposed to compare its value with built-in function `exp(x)`.

Your output should appear in the format mentioned in Table 1.

Task two: A simple calculator

Implement a simple calculator using `Switch` statements. The calculator should perform the operations mentioned in the Table 2.

[1] `http://mathworld.wolfram.com/MaclaurinSeries.html`

Table 1: Output format for the Task one.

n	`Myfunc(x, n)`	error = `exp(x) - Myfunc(x, n)`
1		
⋮		
10		

Table 2: List of operations to be performed by the simple calculator.

Arithmetic	Trigonometric	Others
+	sin()	x^y
-	cos()	$x!$
×	tan()	
/	cot()	

Optional operations: % (modulus operator), $\log_{10}(x)$, $\log_e(x)$, Boolean 'AND' and 'OR'.

Write separate functions for each of the operations. Each function should first check whether the input is valid or not and then carry out the operation.

Task three: Combinatorics (Optional)

Part A: Using the previously written function `Fact(n)`, write the following functions:

`int Perm(int M, int N)`: calculates $\frac{M!}{(M-N)!}$, i.e., all the permutations possible while choosing N items out of M items.

`int Comb(int M, int N)`: calculates $\frac{M!}{N!(M-N)!}$, i.e., all the combinations possible while choosing N items out of M items.

Part B: Find the answers to the following questions using `Perm(M, N)` and `Comb(M, N)` functions you have written in the part A:

1. A box contains 10 black and 6 green marbles. How many ways can 5 black and 4 green marbles be chosen?
2. Out of 4 Women and 5 Men, a committee of 5 is to be formed. In how many ways can it be formed if at least 2 women is to be included?

Some information on exception handling

C functions like `scanf()` return an integer. So far, while calling `scanf`, we have usually not looked at the value it returns. We have been satisfied with its job of reading inputs and copying them into specified memory locations. Now look at the program below, compile it and play with it. In specific while entering l, m please try combinations such as:

56 107, 56, 107, 56.65 107, 56 107.76, -56 107, 56 -107, 5 107, -56 107 etc.

```
1   #include<stdio.h>
2
3   void main(){
4     unsigned int   l, m, n, validEntry=0;
5
6     printf("Please enter a positive integer n, less than 15\n");
7     if ( (scanf("%u", &n) == 1) && (n < 15) ){
8       validEntry = 1;
9     }
10    while (validEntry == 0) {
11      printf("\t You must enter a positive integer less than
          15\n");
12      if ( (scanf("%u", &n) == 1) && (n < 15) ){
13          validEntry = 1;
14      }
15    }
16    printf("Please now enter two positive integers l, m, greater
        than 15\n");
17    n = scanf("%u %u", &l,&m);
18    printf("\t My call of the scanf function to read your two
        numbers l, m returned the value: %u\n",n);
19    if ( ( n == 2) && (l > 15) && (m > 15) ){
20      printf("\t You entered l = %u,\t m = %u.\n",l,m);
21      printf("\t You are a careful user. Goodbye ! \n\n");
22    }
23    else {
24      printf("\t You did not follow instructions carefully. No
          more chances for you !\n\n");
25    }
26  }
```

checkingForValidUserInput.c