MA1700 - Coding and Languages

Homework

Instructions

- Try to solve all problems. They shall be discussed in class on Monday, September 25.
- If you do not understand a term or word in this document, try to search for it online before sending us an email. You might need to be specific in the search terms you use. Using Google search is an art!
- If you find that a problem statement is vague or unclear, feel free to send us an email.

Remarks

- The overall level of this homework is higher than our classes so far. You will have to put in some extra effort!
- If you have questions or comments on the homework, please use the *Reply to all* feature in your email so that your questions/comments are sent to everybody in the class.

Bon courage !

Problem 1 (arrays)

Given two arrays of `float` numbers, calculate their dot product. The dot product of two arrays is defined as the sum of element-wise product of the two arrays. If s and t are two `float []` arrays of size 3, their dot product dp is given by

$$dp = s[0] * t[0] + s[1] * t[1] + s[2] * t[2]. \tag{1}$$

Your program should first ask for the size of the arrays (let us call it n). Your program should then ask for $2n$ `float` numbers, which define the arrays. Finally, your program should print the dot product of these arrays.

Problem 2 (arrays)

Given an array and a prediction of the array (which is an array of the same length), find the mean-squared error between the array and its prediction. If s and t are two `float []` arrays of length 3, then the mean-squared error me between s and t is given by

$$me = \frac{1}{3}\Big((s[0] - t[0])^2 + (s[1] - t[1])^2 + (s[2] - t[2])^2\Big) \tag{2}$$

You can take input in the same manner as Problem 1.

Problem 3 (strings)

Given a number, check whether it is a palindrome or not. A number is a palindrome if it reads the same left-to-right and right-to-left. Two examples of palindromes are 1245421 and 846648. Your program should take the number as input. Print `TRUE` if the number is a palindrome, and `FALSE` otherwise. **Hint**: You can scan the number as a string because there is no need to perform mathematical operations on the number!

Problem 4 (functions)

Write a function that takes two numbers x and y, and an operation *op* (addition, subtraction, multiplication, or division), and returns the result of applying *op* to x and y. *op* is specified using a `char` variable. The function should be defined as

```
float apply_operation(float x, float y, char op)
{
   ... <your definition of the function> ...
}
```

For example, `apply_operation(4.0, 2.0, '/')` should return the value of `4.0/2.0`. **Hint**: Make sure the operation you are trying to perform is valid (division by zero is now allowed!).

Problem 5 (functions)

Write a function that takes a natural number n and returns the n-th Fibonacci number. The n-th Fibonacci number F_n is defined as

$$F_n = F_{n-1} + F_{n-2}, \tag{3}$$

so that $F_2 = F_1 + F_0$, $F_3 = F_2 + F_1$, and so on. The first two Fibonacci numbers are $F_0 = 0$ and $F_1 = 1$. **Hint:** Use a `for` loop in the function.

Problem 6 (structures)

This problem is "more free" with respect to previous problems. You can solve it any way you wish, as long as you use `struct`s and the objective is achieved.

Suppose that you are the instructor of a computer programming course. You want to store data records of your students. There are three things to be stored for each student: the student ID (an `int`), the name (a string), and the points awarded out of 100 (an `int` as well). Create an appropriate structure which will allow you to store this information.

Now, create an array of these structures and populate it with some random data (it could be anything you wish). Your **objective** now is to print the student ID, name, and points of the student who has the maximum points in the class.
