EECE 230 Introduction to Programming, Sections 3,4, and 12
Programming Assignment 6

Thursday Nov 6, 2012

- This programming assignment consists of 3 problems.

  All the problem are syntax oriented: they involve either a a very simple problem, or rewriting a code from a previous Programming Assignment as a function. The objective is to get used to functions and pointers.

- It is due on Tuesday Nov 13 in the Lab.
- Related reading: functions, pointers, passing parameters to functions.
- *Lab structure and regulations:*
  - ⋆ The 3 hours Lab session is on Tuesdays in Lab rooms 1,2, and 5 from 2:00 pm to 5:00 pm. It consists of three parts:
    - *Occasional Solving Session (not graded but attendance mandatory)*
    - *Programming Assignment (graded)*
      Programming Assignments will be posted on Moodle on weekly basis. Typically, a Programming Assignment requires much more than the time allocated for this part in the Lab, so you are supposed to complete the major part of the assignment at home. The Lab instructor will grade your assignment and can help you with the problems you are facing.
    - *Occasional graded weekly quiz*
  - ⋆ You are supposed to submit your own work. Cheating will not be tolerated and will be dealt with severely: zero grades on the programming assignments, disciplinary committee, Dean's warning.
  - ⋆ Lab attendance is mandatory. Violating this rule can lead to a failing grade.

Problem 1 (Circles: functions warmup)

Do Programming Exercise 6.6 [Malik, page 343 in the third edition]:

The following formula gives the distance between two points (x_1, y_1) and (x_2, y_2) in the Cartesian plane:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Given the center and a point on the circle, you can use this formula to find the radius of the circle. Write a program that prompts the user to enter the center and a point on the circle. The program should then output the circle's radius, diameter, circumference, and area. Your programs must have at least the following functions:

a) *distance:* This function takes as its parameters four numbers that represent two points in the plane and returns the distance between them.

b) *radius:* This function takes as its parameters four numbers that represent the center and a point on the circle, calls the function distance to find the radius of the circle, and returns the circle's radius.

c) *circumference:* This function takes as its parameter a number that represents the radius of the circle and returns the circle's circumference. (If r is the radius, the circumference is $2\pi r$.)

d) *area:* This function takes as its parameter a number that represents the radius of the circle and returns the circle's area. (If r is the radius, the area is πr^2.)

Assume that $\pi = 3.1416$.

Problem 2 (Selection Sort revisited)

a) **Array Print function:** Write the function *arrayPrint* which takes as its parameters (a pointer to) an array of integers, and its size. It prints the array.

b) **Array Smallest function:** Write a function *arraySmallest* which takes as parameters (a pointer to) an array A of integers and two indices *start* and *end* of A (assume $0 \leq start \leq end$). It supposed to find and return the index of the smallest element in $A[start...end]$.

c) **Selection Sort function:** Using the *ArraySmallest* function in part (b) and the *swap* function we did in class, write the selection sort function *selectionSort* which takes as its parameters (a pointer to) an array of integers, and its size. This function sorts the array using the Selection Sort algorithm [Problem 4 of Programming Assignment 4].

Using those functions, you are supposed to get a very simple and structured code for the selection sort algorithm.

Look first at the solution of Problem 3 in PA 3.

d) **Test Program:** Write the following program to test the above functions.

```
prototype of printArray
prototype of arraySmallest
prototype of swap
prototype of selectionSort

main function:
 1.  ask the user to an enter an integer n
 2.  allocate memory for an array A of integers of size n
 3.  fill the array with the user input
 4.  call  the printArray function to print A
 5.  call the selectionSort function to sort A
 6.  call the printArray function to print the sorted version of A
 7.  free the memory allocated for the array A
end of main

bodies of the above 4 functions
```

Problem 3 (String functions)

a) **Deleting a character from a string revisited:** Write the function *myStringDeleteChar* which takes as input parameters a C-string *str* and an integer i.

It is supposed to delete the the i'th character of s, where $i = 0$ corresponds to the first character, $i = 1$ corresponds to the second character, and so on ....

Moreover, *myStringDeleteChar* is supposed to return the value of the deleted character.

If $i < 0$ or $i \geq length(s)$, your function is supposed to return the null character $'\backslash 0'$ without modifying the string.

You can use the *strlen* function.

Use the following program to test your functions:

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

// myStringDeleteChar  prototype

int main()
{
      char s[] = ``eece230'';
      char c = myStringDeleteChar(s,3);
      cout <<c<<endl;
      cout <<s<<endl;
      return 0;
}
// myStringDeleteChar  body
```

You are supposed to get

```
e
eec230
```

Use the code from the solution of Problem 2 of PA 4 and appropriately put it in the body of a function.

b) **Pointers version:** Repeat part (a) without using the array subscription operator [] at all in the prototype or body of the *myStringDeleteChar* function. Use instead the pointer declaration operator * and the indirection operator *.

c) **Remove duplicate spaces function**

Write a function *removeDuplicateSpaces*, which given two *C*-strings $s1$ and $s2$, removes the duplicate (consecutive) spaces from $s1$ and stores the resulting string in $s2$. Assume that enough memory is allocated to $s2$ before calling the function. For simplicity, you can assume that $s1$ does not contain tabs.

For instance if $s1$ contains

```
Write  a   function      removeDuplicateSpaces,     which   given two   C-strings
```

the function is supposed to store in $s2$:

```
Write a function removeDuplicateSpaces, which given two C-strings
```

Use the following test program

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

// removeDuplicateSpaces prototype

int main()
{
     char s1[500], s2[500];
     cout << ``Enter string:'';
```

```
     cin.get(s1,500);// read a line (possibly containing spaces) of length
                     // at most 499 characters and strote it into s1.
     removeDuplicateSpaces(s1,s2);
     cout << ‘‘Duplicate spaces removed:’’<<endl;
     cout<< s2 <<endl;
     return 0;
}

// removeDuplicateSpaces body
```