# CSE102 – Computer Programming

# Homework #8

## Due Date: 29/04/2020

**Hand in:** A student with number 20180000001 should hand in a 'c' file named 20180000001.c for this homework.

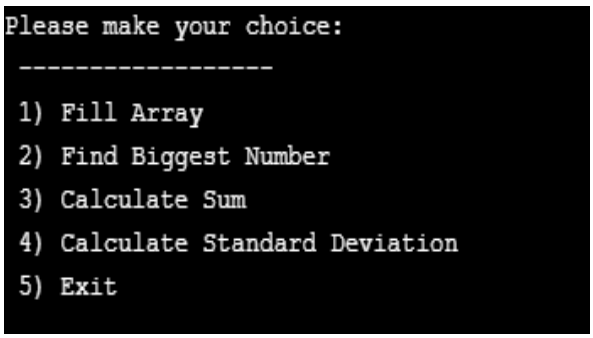
---

**Part 1.** [40 pts] Lets do some exercises to reinforce your recursion skills. You are asked to write a function that allows to do some useful operations on Hofstadter's Q-Sequence. The formula of these sequences is shown below:

$$Q(n) = Q\left(n - Q(n-1)\right) + Q\left(n - Q(n-2)\right)$$

$$Q(1) = Q(2) = 1$$

You must write four operational functions and one menu function to successfully do this part. Menu function should look like the following:




The function takes an empty, fixed-sized (define the size of the array as 500 and use it where you needed) integer array that is used by the array operations in the menu:

- Fill array: The option takes an integer number (n) from the user and fills the array with first n items of the Hofstadter's Q-Sequence. The array items that out of the range (n) should be zero. The function prototype of this part is shown below:

```
void generate_hofstadters_sequence (int *arr, int n)
```

- Find biggest number: The function finds the maximum valued item in the array recursively. The function prototype of this part is shown below:

```
int find_max (int arr[], int index, int max_value)
```

- Calculate sum: The function calculates the sum of items in the array recursively. The function prototype of this part is shown below:

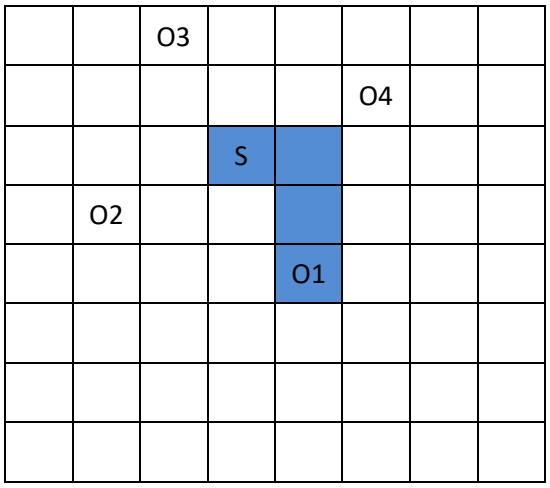
```
int sum_array (int arr[])
```

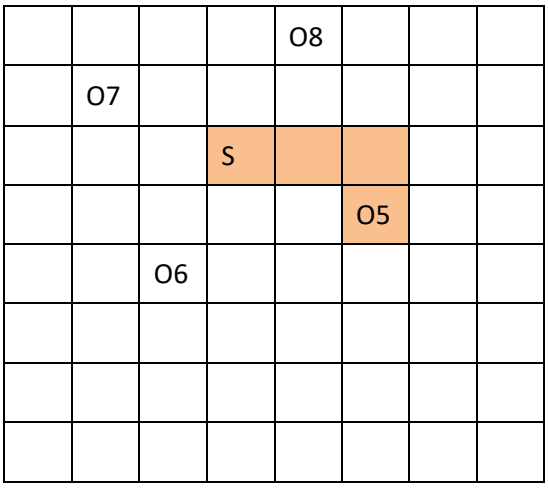
- Calculate standard deviation: The function calculates the mean and the standard deviation of the array recursively. The function returns standard deviation by default and the mean value returns by reference.

```
double std_array (int arr[], double *mean, int n, int index)
```

- Exit: Terminates the menu function. The menu should work until user select this option. Don't forget to print the calculated/found values after the recursive functions done their jobs on option 2,3 and 4.

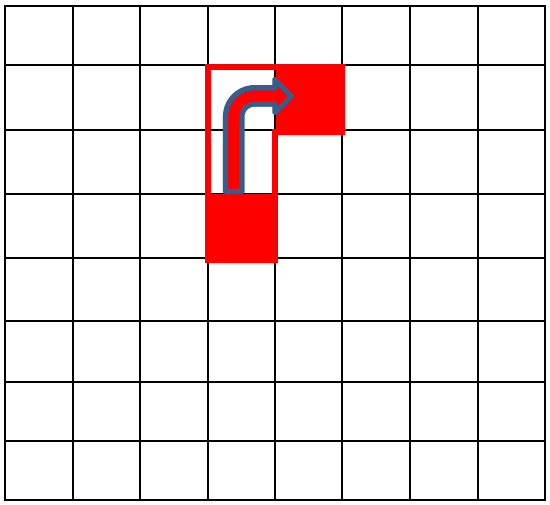
**Part 2.** [30 pts] Assume that we have a square field consisting of 8x8 blocks as shown below:

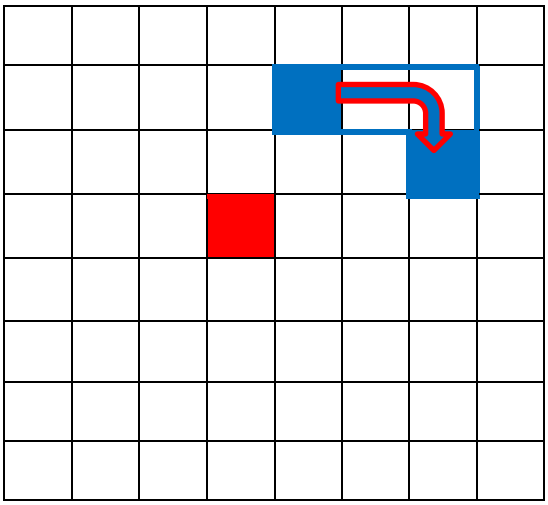



We want to install pipes for an irrigation system on this field. We have only an L-shaped pipes (covering 3 blocks in one direction and 2 blocks in the perpendicular direction) that can be connected to cover any given part of the field. There are 8 possible placements of an L-pipe starting from a given location. When starting from the block "S", an L-shaped pipe can reach to the blocks labeled "O1, O2, O3, O4, O5, O6, O7, O8 " as shown in the figure above. Each of these "O?" labels indicate a unique orientation of the pipe placement.

Assume that the field is represented by an integer array of dimensions 8x8. Initially this array has all 0's indicating there are no pipes placed. The action "At location [3,3] place an L pipe with orientation 4" will fill in the field as shown on the left below. Another action "At location [1,4] place an L pipe with orientation 5" will fill in the field as shown on the right below.





Write a recursive function that prints all possible orientation sets that covers the entire field with L-shaped pipes (starting from the **top**-left corner of the field). Remember that one action covers only one field at a time and the action can't be done if the target block is already visited before. The function prototype has given below:

```
void install_pipes (int visited[N][N], int x, int y, int orientation)
```

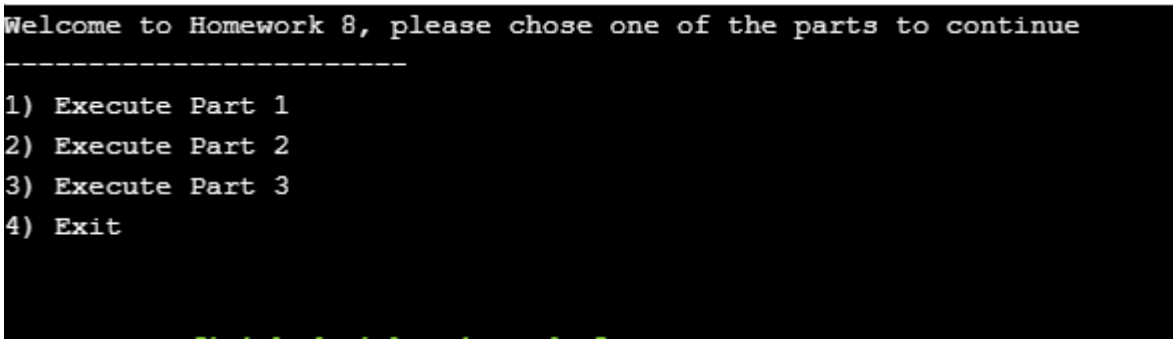
where orientation is the start orientation of the first L-shaped pipe, x and y are the coordinates of the start position and visited is the field. An example output should be like the following:

O1, O4, O6, O2 …

**Part 3.** [25 pts] Write a recursive function that removes duplicated characters in a given string. For example; Let the given string is "Thiis is not completely misspelled butt we wannnnt to fix it". The string becomes "This is not completely mispeled but we want to fix it" after the function was performed on it. Get the string from user. You can specify a maximum length for the input string. The function prototype is given as:

```
char* remove_duplicates (char* str)
```

**Part 4.** [5 pts] Write a menu function that allows to test all the three parts. Menu function should work until user wants to exit. The menu should seem as below:




The function prototype is:

```
void menu ()
```

**General Rules:**

1. Obey and do not break the function prototypes that are shown on each part, otherwise, you will get zero from the related part.
2. The program must be developed on Linux based OS and must be compiled with GCC compiler, any problem which rises due to using another OS or compiler won't be tolerated.
3. Note that if any part of your program is not working as expected, then you can get zero from the related part, even it's working in some way.
4. Upload your .zip file on to Moodle to deliver your homework. The zip file must consist of three .c file that contains your solutions. Name format can be found on the top of this homework sheet.
5. You can ask any question about the homework by sending an email to b.koca@gtu.edu.tr or by using the forum in the Moodle page of the course.