MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG140 C Programming Assignment 2

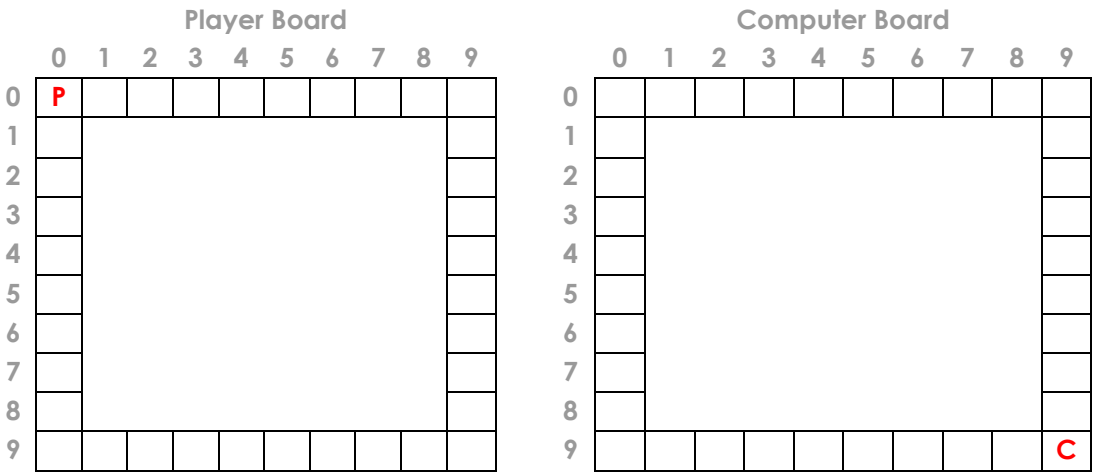
Date handed out: **25 April 2024**
Date submission due: **09 May 2024 23:55**

Programming Assignment 2: Circle

Purpose:
The main purpose of this programming assignment is to revise the topics that we have covered so far including arrays, pointers, functions, repetitive statements, conditional statements, and fundamentals of C programming. In this assignment, you are going to implement a two-player Circle game based on the given specifications below.

Do not try to compile your entire program in one "big bang". Compile it piece by piece. **Test each function/piece that you have compiled to make sure it works correctly before you add the next function/piece.**

Game Specifications:
You will write a program for playing a Circle game between a player and a computer. The objective of the game is to move around the board to complete a circle. The game will start with two boards of 10x10 which means there will be ten rows and ten columns. The first index of the first board (0,0) will be the starting point for the player and initialized as 'P'. The last index of the second board (9,9) will be the starting point for the computer and initialized as 'C'. The boards below show an illustration of the initial boards.



Then, the player and computer will choose a trap index. The trap index for the player should be entered by the user while it should be randomly generated for the computer. Then, a value between 1 and 5 will be randomly assigned to the trap index for both the player and the computer. The trap index can be only placed on the around of the board. For example for player it could be any of these indexes: (0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7),(0,8),(0,9),(1,9), (2,9),(3,9), (4,9),(5,9), (6,9),(7,9), (8,9),(9,9), (9,8), (9,7), (9,6), (9,5), (9,4), (9,3), (9,2), (9,1), (9,0), (8,0), (7,0),(6,0),(5,0),(4,0),(3,0),(2,0),(1,0). See below example;

```
----Player----
Enter trap index (row col): 0 5
Generated trap value: 2
----Computer----
Generated trap index (row col): 9 5
Generated trap value: 3
```

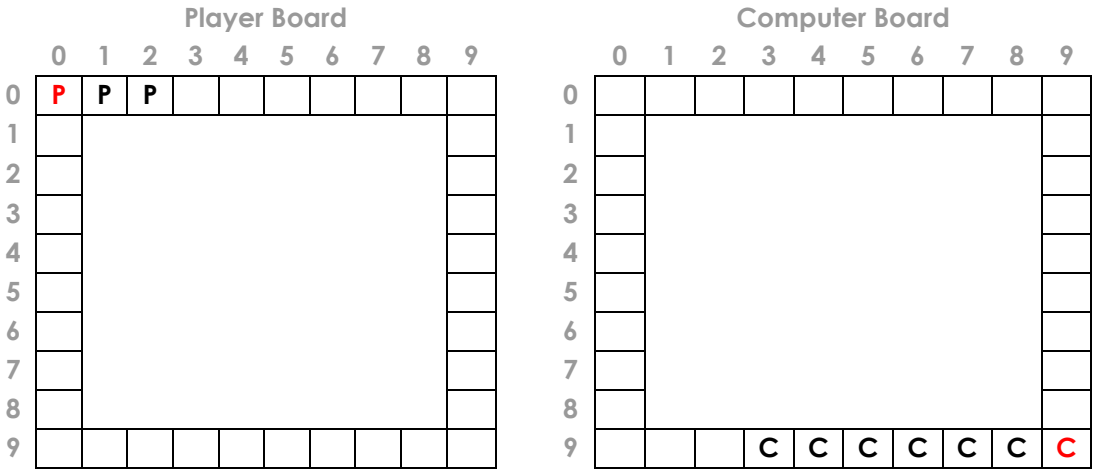
If the user enters a wrong trap index, then your program should give an error message and ask the user to enter the trap index again. See below example;

```
----Player----
Enter trap index (row col): 1 5
The trap index should be placed around the board.
Enter trap index (row col): 5 7
The trap index should be placed around the board.
Enter trap index (row col): 0 9
Generated trap value: 2
----Computer----
Generated trap index (row col): 0 2
Generated trap value: 4
```

Then, to decide who is going to start the game, each player rolls one die. The highest roll goes first, roll a die, and take that amount of step in a clockwise direction around the board. See below example;

```
I have rolled the dice and got 6!
I have rolled the dice for you and you got 5!

----Round 1----
I have rolled the dice and got 6!
I have rolled the dice for you and you got 2!
```



If the movement stops at the trap index, then it should calculate the amount of step, which will be the amount of step forward – trap value. If the result is less than or equal to zero, then there

will not be any movement. If the movement passes through the trap index and stops at some other index then it can move, it's not trapped.

The game will finish whenever the player or the computer completes a circle (takes steps from the starting point, moves clockwise around the board, and reaches the starting point (which is (0,0) for the player and (9,9) for the computer). The boards below show an example illustration of the final boards where the player completes the circle and wins the game.

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1	P									P
2	P									P
3	P									P
4	P									P
5	P									P
6	P									P
7	P									P
8	P									P
9	P	P	P	P	P	P	P	P	P	P

Computer Board

	0	1	2	3	4	5	6	7	8	9
0	C	C	C	C	C					
1	C									
2	C									
3	C									
4	C									
5	C									
6	C									
7	C									
8	C									
9	C	C	C	C	C	C	C	C	C	C

A possible sample run could be as follows:

Please note that the game is told from the computer's perspective in the sample run

```
Welcome to the Circle Game! :)
Let's get started!

----Player----
Enter trap index (row col): 1 5
The trap index should be placed around the board.
Enter trap index (row col): 1 7
The trap index should be placed around the board.
Enter trap index (row col): 0 9
Generated trap value: 2
----Computer----
Generated trap index (row col): 8 0
Generated trap value: 4

I have rolled the dice and got 6!
I have rolled the dice for you and you got 5!
```

```
----Initial Boards----
```

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P									
1										
2										
3										
4										
5										
6										
7										
8										
9										

Computer Board

	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5										
6										
7										
8										
9										C

```
----Round 1----
I have rolled the dice and got 2!
I have rolled the dice for you and you got 6!
```

Player Board

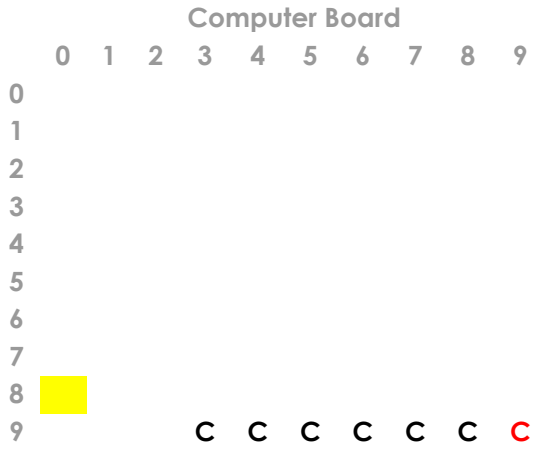
	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P			
1										
2										
3										
4										
5										
6										
7										
8										
9										

Computer Board


```
----Round 2----
I have rolled the dice and got 4!
I have rolled the dice for you and you got 3!
Trapped at index 0 9! 3 forward - 2 backward = 1 step.
```

Player Board

Computer Board



```
----Round 3----
I have rolled the dice and got 4!
Trapped at index 8 0! 4 forward - 4 backward = 0 step.
I have rolled the dice for you and you got 6!
```

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1										P
2										P
3										P
4										P
5										
6										
7										
8										
9										

Computer Board

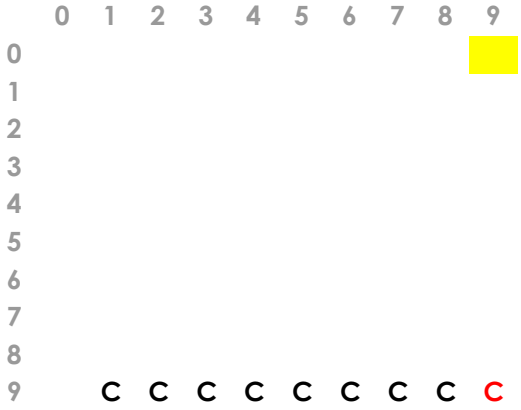
	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5										
6										
7										
8										
9				C	C	C	C	C	C	C

```
----Round 4----
I have rolled the dice and got 2!
I have rolled the dice for you and you got 5!
```

Player Board

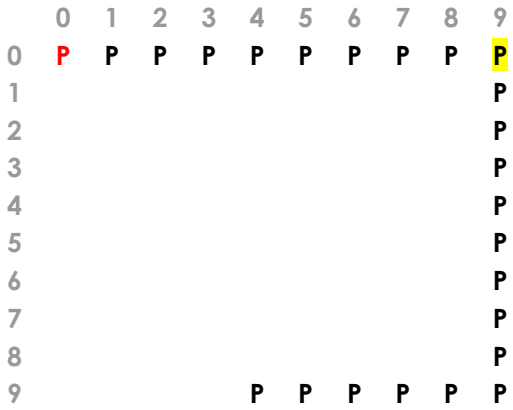
	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1										P
2										P
3										P
4										P
5										P
6										P
7										P
8										P
9										P

Computer Board




```
----Round 5----
I have rolled the dice and got 2!
Trapped at index 8 0! 2 forward - 4 backward = 0 step.
I have rolled the dice for you and you got 5!
```

Player Board



Computer Board

```
----Round 6----
I have rolled the dice and got 6!
I have rolled the dice for you and you got 4!
```

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1										P
2										P
3										P
4										P
5										P
6										P
7										P
8										P
9	P	P	P	P	P	P	P	P	P	P

Computer Board

	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4	C									
5	C									
6	C									
7	C									
8	C									
9	C	C	C	C	C	C	C	C	C	C

```
----Round 7----
I have rolled the dice and got 6!
I have rolled the dice for you and you got 6!
```

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1										P
2										P
3	P									P
4	P									P
5	P									P
6	P									P
7	P									P
8	P									P
9	P	P	P	P	P	P	P	P	P	P

Computer Board

	0	1	2	3	4	5	6	7	8	9
0	C	C	C							
1	C									
2	C									
3	C									
4	C									
5	C									
6	C									
7	C									
8	C									
9	C	C	C	C	C	C	C	C	C	C

```
----Round 8----
I have rolled the dice and got 5!
I have rolled the dice for you and you got 6!
```

Player Board

	0	1	2	3	4	5	6	7	8	9
0	P	P	P	P	P	P	P	P	P	P
1	P									P
2	P									P
3	P									P
4	P									P
5	P									P
6	P									P
7	P									P
8	P									P
9	P	P	P	P	P	P	P	P	P	P

Computer Board

	0	1	2	3	4	5	6	7	8	9
0	C	C	C	C	C	C	C	C		
1	C									
2	C									
3	C									
4	C									
5	C									
6	C									
7	C									
8	C									
9	C	C	C	C	C	C	C	C	C	C

```
----END----
You won!!!

Would you like to play again (Y/N)? N
```

Programming Requirements:
To implement this game, you will need to write at least the following functions, but if you need more functions you can add them.

Function	Explanation
`main`	The main function will create the player and computer boards of 10x10, and these boards will be maintained here. Initially, the boards are empty with the initialized 'P' and 'C' starting points. Trap row and column index will be taken as input from the user for the player and `checkTrapIndex` function will be called to check if the user placed the trap index around the board or not. If not, it will show the appropriate message and ask the user to enter again. `generateTrapIndex` function will be called to generate trap row and column index for the computer. `generateTrapValues` function will be called to generate trap values for the player and the computer. Then, `rollDice` function will be called to decide which player will start the game first. Then accordingly, each of them rolls a dice and then `stepPlayer` and `stepComputer` functions will be called until any of them completes the circle. At the end of the game, it will print who wins the game and ask if they would like to play again or not.
`stepPlayer`	This function will take the array and step number and place the 'P' to the next available empty locations. Should also calculate the step number considering the trap value. Whenever the circle is completed, it should return 1, otherwise it should return 0. **Input**: array of the board, number of rows and number of columns, a step number, trap row index, trap column index, trap value **Output**: 0 or 1
`stepComputer`	This function will take the array and step number and place the 'C' to the next available empty locations. Should also calculate the step number considering the trap value. Whenever the circle is completed, it should return 1, otherwise it should return 0. **Input**: array of the board, number of rows and number of columns, a step number, trap row index, trap column index, trap value **Output**: 0 or 1

`checkTrapIndex`	This function will take the row and column index of the trap index for the player, and if it is correct (trap index can be only placed on the around of the board, see possible indexes given on page 1) it will return 1 and return 0 otherwise. **Input**: row index, column index **Output**: 0 or 1
`generateTrapIndex`	This function will generate the row and column index of the trap index for the computer. **Input**: row index, column index **Output**: none
`generateTrapValues`	This function will generate the trap values for the computer and the user. **Input**: trap value for the player, trap value for the computer **Output**: none
`rollDice`	This function will generate a random number between 1 and 6. **Input**: none **Output**: a random number which is between 1 and 6
`displayBoard`	This function takes the board and displays it on the screen. **Input**: array of the board, number of rows and number of columns **Output**: none

Grading Policy:

If your code does not compile, you will automatically get zero. If your code compiles, you will then be graded based the following scheme:

Grading Point	Mark (100)
`Main`	30
`rollDice`	5
`stepPlayer`	15
`stepComputer`	15
`checkTrapIndex`	10
`generateTrapIndex`	5
`generateTrapValues`	5
`displayBoard`	5
Code quality (e.g., formatting, commenting, naming variables, clean use of C constructs such as formulation of selection statements and loops, etc)[1]	10

[1] See guidelines given here: https://www.gnu.org/prep/standards/html_node/Writing-C.html

If you do not obey the following rules then you will automatically get 0.

Rules:

Please make sure that you follow the restrictions for the assignment as follows.

- **Strictly obey the input-output format. Do not print extra things.**
- **You are NOT allowed to use global variables and goto statements.**
- **You need to organize your code in the given functions, however, if you do not have functions, then your grade will be out of 50%.**
- **Add your name/surname and ID at the top of your code as comments and name your source file "main.c".**
- **Submit your "main.c" to odtuclass. Do not compress your project or submit other files (zip, rar, …). You only need to submit "main.c".**

Cheating:

Full description of cheating, please refer to https://ncc.metu.edu.tr/res/academic-code-of-ethics, but here we provide a description for assignments in this course. The following will be considered as cheating and the Professionalism and Ethics rules given in the course syallbus will be applied accordingly:

1. Students are expected to complete the assignments on their own. Working with other students in assignments teamwork is NOT allowed (Working together is limited to brainstorming only. The "effort" in the assignments should belong completely to the student);
2. Submitting assignments, taken partially or entirely from others or from a source (book, internet, paper, etc);
3. Submitting assignment/exam, taken partially or entirely from a previously graded work;
4. Using generative AI tools to generate solutions and/or presenting generative AI solutions as one's original work will also be considered cheating.