MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG140 C Programming Assignment 3

Date handed out: **10 May, Monday**
Date submission due: **30 May, Sunday 23:55**

Programming Assignment 3: Shoot and Merge

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

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

Game Specification:
This game will start with an empty board of 5x4 which means there will be five rows and four columns. The game will then generate a random number which is the power of two. The user then will decide where to shoot this number. The user can shoot it from the bottom of one of the columns and then if there is already a number with the same value then it will be merged. The game will be finished when the board is full. The following sample run aims to illustrate the game.

The game starts with an empty board as follows:

A random number is generated by the game, let's assume it is 2. Then the user will decide where to shoot it. Let's say the user shoots it to column 1. Then the board will become as follows:

2			

A random number is generated by the game, let's assume it is 4. Then the user will decide where to shoot it. Let's say the user shoots it to column 2. Then the board will become as follows:

2	4		

A random number is generated by the game, let's assume it is 4. This time the board has the same number in column 2 therefore the user can shoot it there to merge it. Therefore, if the user chooses column 2 then it will merge it and here is the board after this shoot.

2	8		

The goal of the player is to maximise the merge. When the board is full that means there is no possibility for merge, the game will finish and it will report user the number of shoots that were made until the board is full.

Programming Requirements:
In order 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 board of 5x4, and this board will be maintained here. Initially the board is empty and a random number can be generated up to 32. Therefore, it will call the randomPowerTwo with 32, and it will start the game by calling the function shootMerge. Please note that shootMerge function returns a new maximum value, therefore your maximum value should be updated accordingly. After shootMerge, it will call the fullorNot function to see if the array is full, if it is not then will continue to call shootMerge function. Once the board is full, the main should ask user if they would like to play again. Please note that the main needs to also maintain the number of shoots which will be reported once the game finishes.
`randomPowerTwo`	This function will generate a random number which is the power of two. It will take a maximum value and then generate a number which can be up to that number. For example, if the maximum is 64 which is 2^6 means this function can generate a number between 2^0 and 2^6. To be more specific, it can be one of these: 2^0, 2^1, 2^2, 2^3, 2^4, 2^5 or 2^6. **Input**: max value. **Output**: a random number which is between 2^0 and max
`shootMerge`	This function will take the array, a random number and then will ask user where to shoot. Once the user shoots, and if there is a possibility to merge, the array will be updated accordingly. Lets consider different cases: <u>Case 1:</u> if the column is as follows:

	2 And 2 is shot then this value will become 4. 4 <u>Case 2:</u> However, if there is a column which is as follows: 4 2 And the value shot is 2 then the merge will be as follows: 4 4 Then it will be merged as 8, and the result will be 8. Case 3: If the column is as follows: 4 8 16 And the value shot is 4 then the column will be as follows: 4 8 16 4 If there is a merge, this function needs to also update the maximum number in this array. **Input**: array of the board, number of rows and number of columns, a random number **Output**: maximum value in the array.
`displayBoard`	This function takes the board and displays it on the screen. It should only display the parts of the board that is not empty. **Input**: array of the board, number of rows and number of columns **Output**: none
`fullorNot`	This function takes the board and checks if it is full. If it is then it will return 1, otherwise it will return 0. **Input**: array of the board, number of rows and number of columns **Output**: 1 if the board is full, 0 if the board is still not full

Then Sample Run could be as follows:

```
*Shoot Merge*
Lets get started!
     C1   C2  C3  C4

You have 2, which column would you like to shoot [1-4]?1
     C1   C2  C3  C4
     2
1 shot so far!

You have 4, which column would you like to shoot [1-4]?2
```

```
     C1   C2  C3  C4
     2    4
2 shot so far!

You have 4, which column would you like to shoot [1-4]?5
Sorry, that is not a valid column, try again!
You have 4, which column would you like to shoot [1-4]?2
     C1   C2  C3  C4
     2    8
3 shot so far!

You have 8, which column would you like to shoot [1-4]?2
     C1   C2  C3  C4
     2    16
4 shot so far!

You have 8, which column would you like to shoot [1-4]?3
     C1   C2  C3  C4
     2    16  8
5 shot so far!

You have 4, which column would you like to shoot [1-4]?4
     C1   C2  C3  C4
     2    16  8   4
6 shot so far!

You have 2, which column would you like to shoot [1-4]?1
     C1   C2  C3  C4
     4    16  8   4
7 shot so far!

You have 2, which column would you like to shoot [1-4]?1
     C1   C2  C3  C4
     4    16  8   4
     2
8 shot so far!

You have 2, which column would you like to shoot [1-4]?1
     C1   C2  C3  C4
     8    16  8   4
9 shot so far!


…….
The board is full now with 100 shots!
Game over!
Would you like to play again (Y/N)? T
```

```
That is not valid, please try again!

Would you like to play again (Y/N)? Y
Lets get started!
     C1   C2  C3  C4
……
The board is full now with 200 shots!
Game over!
Would you like to play again (Y/N)? N
Byeee!
```

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
`randomPowerTwo`	10
`shootMerge`	30
`displayBoard`	10
`fullorNot`	10
Code quality (e.g., formatting, commenting, naming variables, clean use of C constructs such as formulation of selection statements and loops, etc)[1]	10

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.**
- Add your name/surname and ID at the top of your code as comments and name your source file "Name-Surname-StudentID.c"
- Submit your solution as C and PDF to odtuclass. Do not compress it (zip, rar, …).

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