2048 GAME

BY:

MEENAKSSHI S

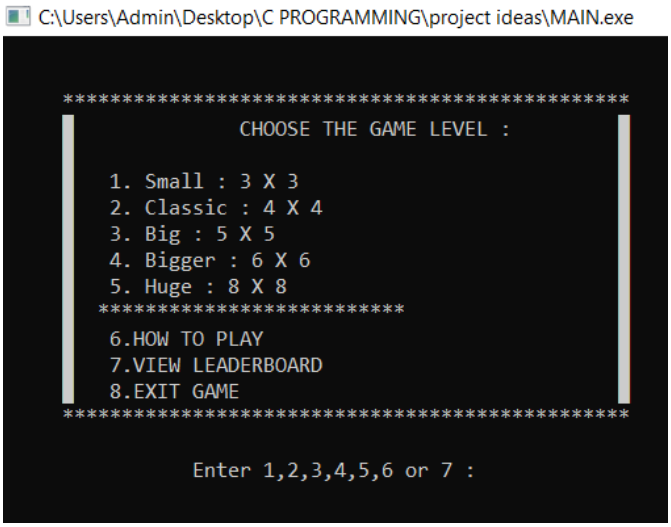
CB.EN.U4CSE19131

- *I have created and implemented a single player sliding block puzzle game*
  - ✓ *Language used: C Programming*
  - ✓ *In addition to the classic 2048 game which is played on a 4x4 grid, this game contains several levels with grid sizes of 3,4,5,6 and 8. So the player has an option to choose the level based on the complexity of the game.*
  - ✓ *Every time a user plays this game, the user's details along with their score is saved on the leader-board*
  - ✓ *Outcome - main Concepts implemented: Structure, file handling, pointers, 2D array, Bubble sort.*
- The game's objective is to slide numbered tiles on a grid to combine them to create a tile with the number 2048.

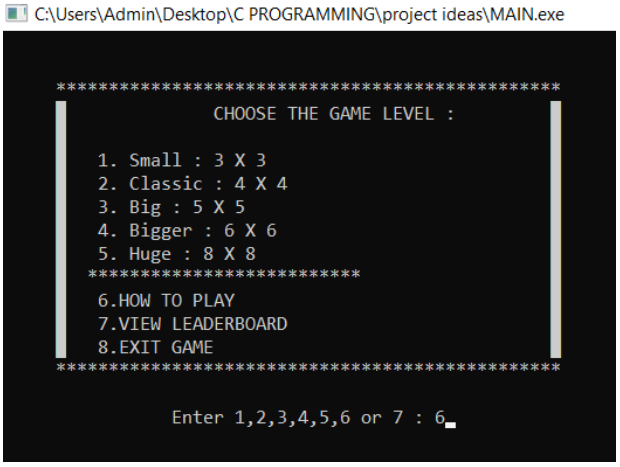
***I HAVE ATTACHED THE SOURCE CODE AND EXECUTABLE C PROGRAM (OUTPUT CONSOLE) ALONG WITH THIS WORD DOCUMENT.**

HERE IS MY EXPLANATION TO THE CODE ALONG WITH THE OUTPUT SCREEN :

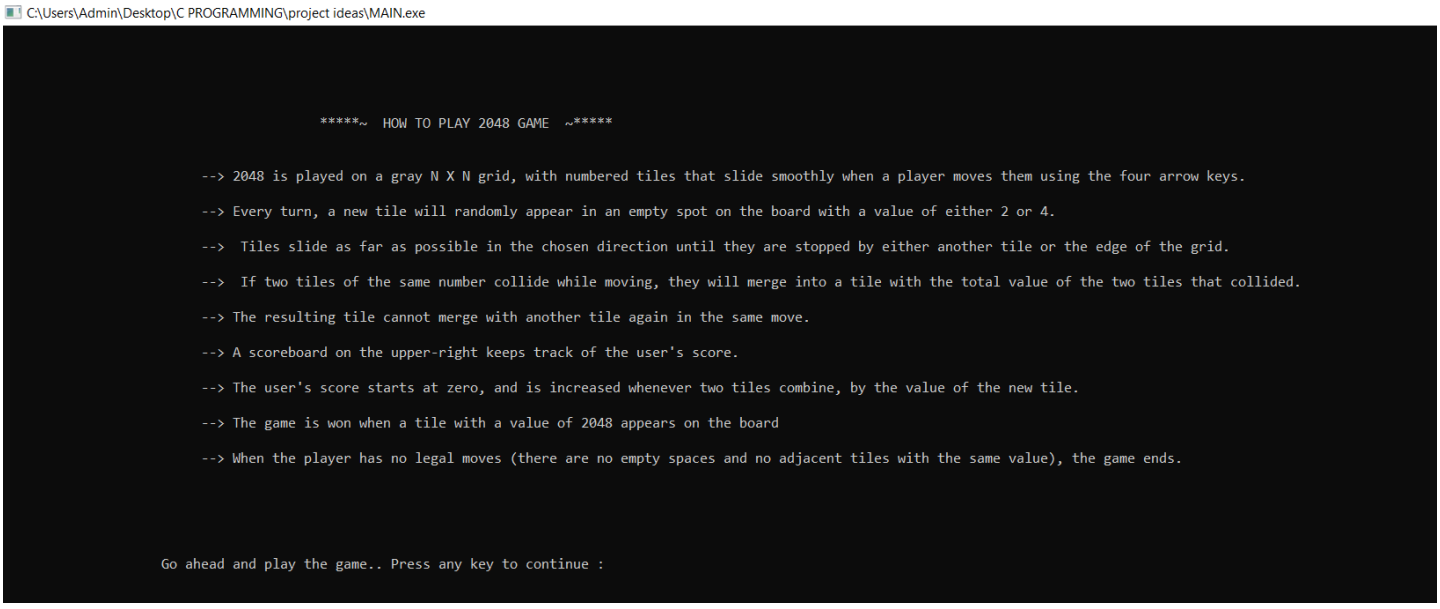
- This is the screen that opens when the application is launched.



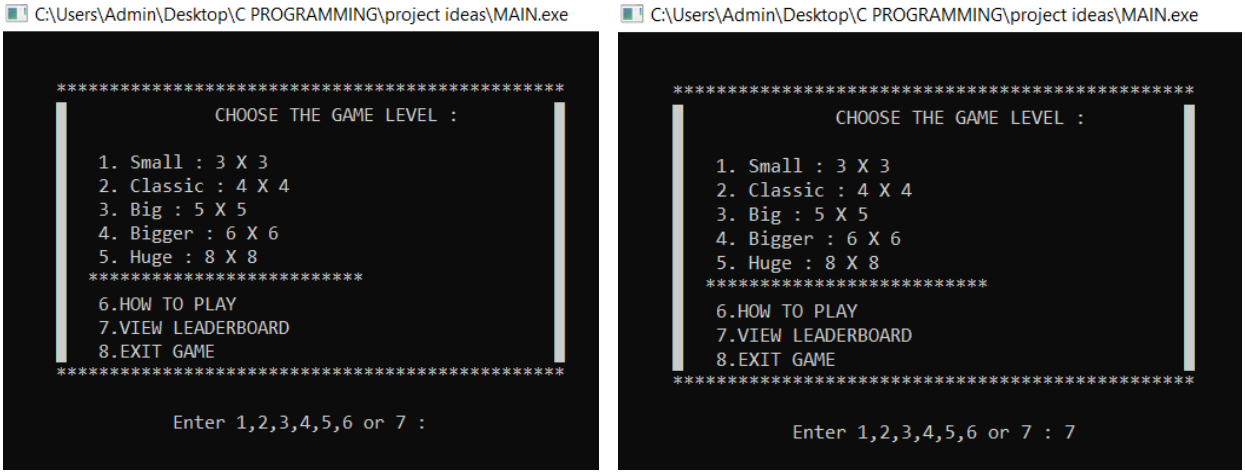
- I have used Switch control statement to dispatch execution to different parts of code.
- Now I will explain the last three options before going to the main part of the game:
- Here, I enter the value 6



->*THE INSTRUCTIONS SCREEN OPENS.*

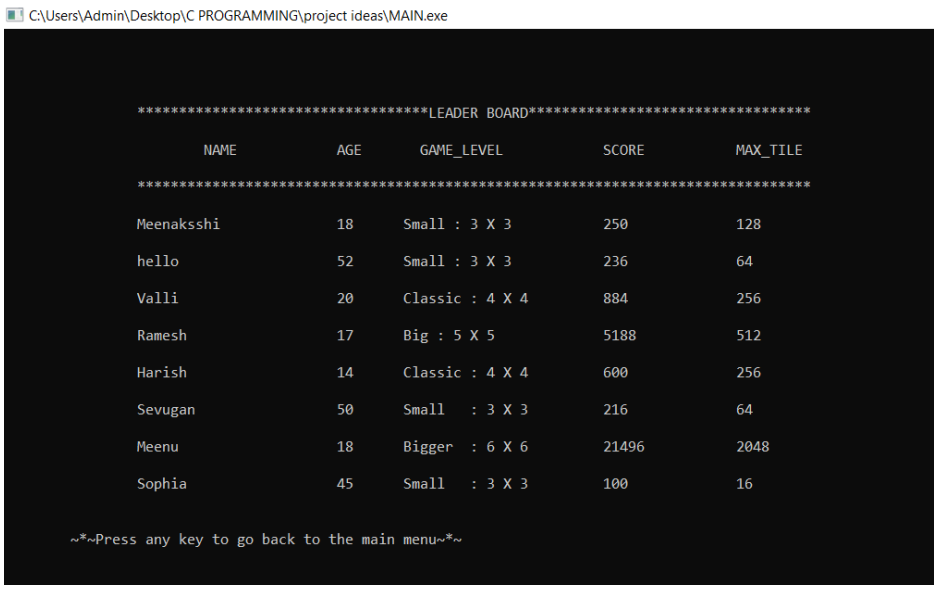


- Here I have given detailed instructions for the users who haven't played the game before.
- I have used 'system("CLS");' function to clear the previos screen and display the new screen.
- Gotoxy(int x, int y) function is used to move the cursor to the desired location on the output console.
- This page redirects the user to the main home page after the user clicks any key.



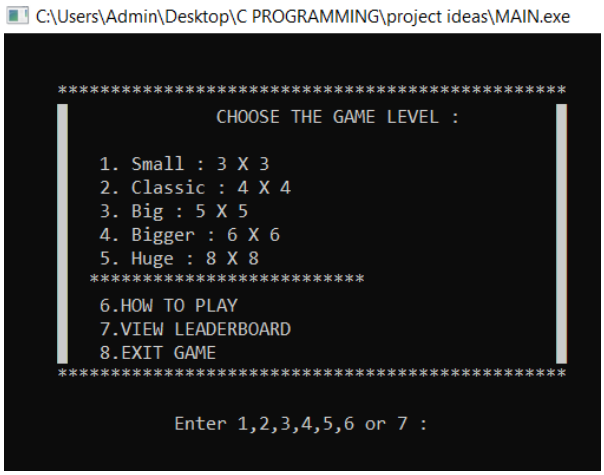
- Now, I will enter 7 to display the Scrore Board

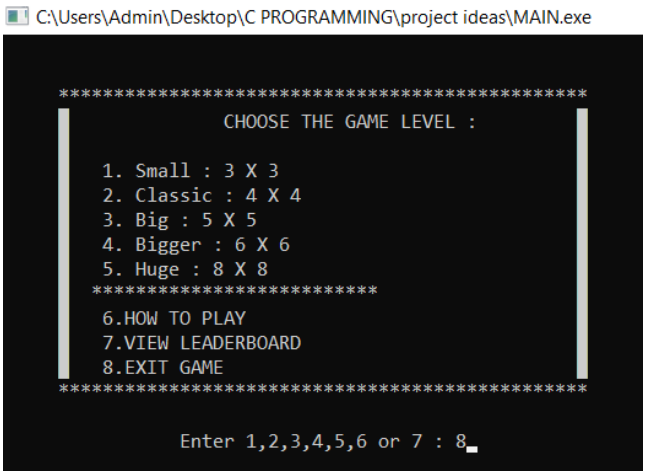
- The LeaderBoard screen opens where the scores and the other details of the previous players are displayed.



- ✓ I have used **Structures** and **file** concept to display this scoreboard.
- ✓ The details of the players are stored in a file using **fprintf** each time a game ends.. and **fgets** is used to scan the value inside the file and diplay it in the output screen.

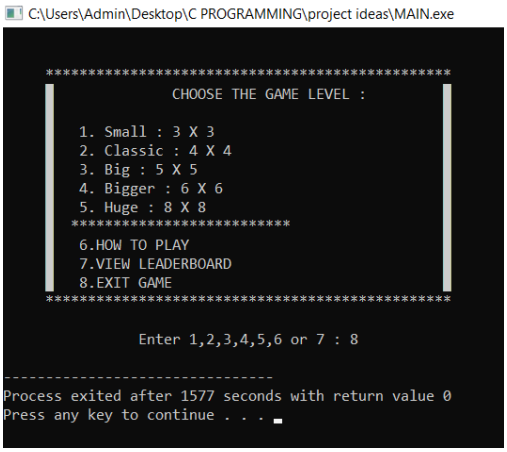
- ✓ Each row corresponds to the value stored in the data members of the structure.
- ✓ This page redirects to the main home page after the user clicks any key.





- ➢ Now, I will enter 8 to exit the game.

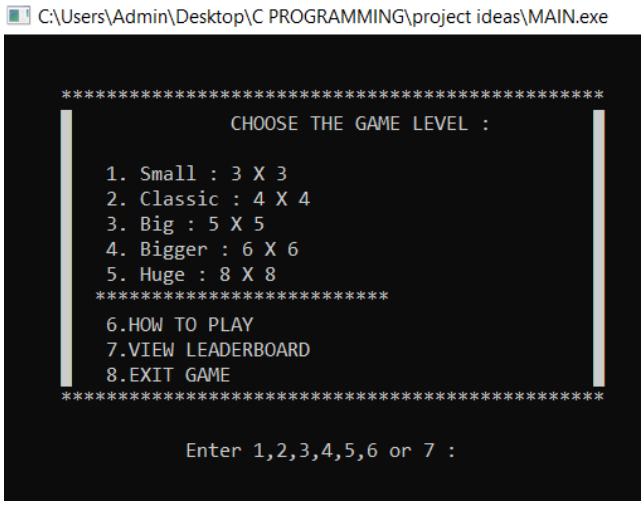
If the player wants to quit the game, 8 is enterd.



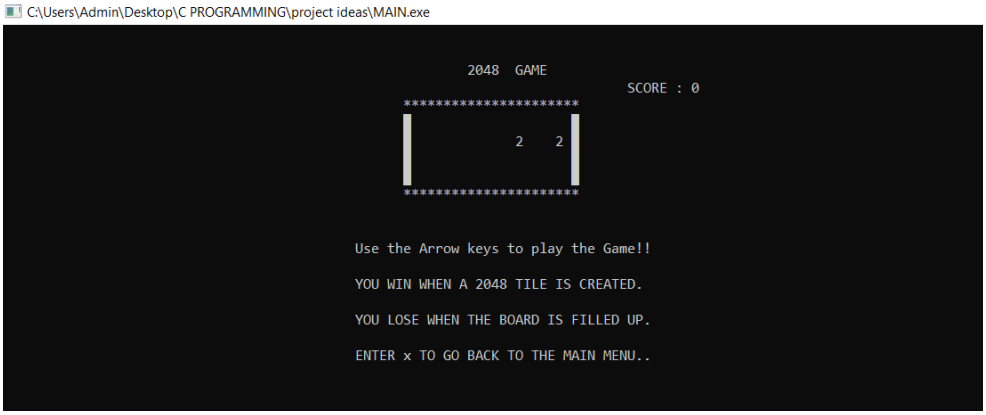
I have used exit(0); to terminate the program.

Lets go to the main part of the game..

- ❖ I have given 5 options for the player to choose his level based on the coplexity of the game.



I will explain this game using the 2nd option: CLASSIC MODE (4 X 4) :



- The main data structure using which this game is implemented is multidimensional array - 2D array.
- First I have initialised all the values in the 2D array to 0.
- Wherever there is a zero “ ” (space) is printed to avoid confusion in the game.
- Then 2 random tiles in the game are filled with either 2 or 4.
- This is done using :

```
//to choose a random tile:
int randomno(int size)
{
    return rand()%size;
}
//90% of the times 2 is added and only 10% of times 4 is added to the 2D array.. this is the rule of the game.
```

```
int randomvalue()
{
        return rand()%10?2:4;
}
```

//This is done **using <u>ternary operator</u>**.

```
x=randomno(size);
y=randomno(size);
if(tile[x][y]==0)
        tile[x][y]=randomvalue();
```

//This is how the random value is added to a random empty tile in the 2D array.

- Now, if I click the right arrow, both the 2's are added up to 4,

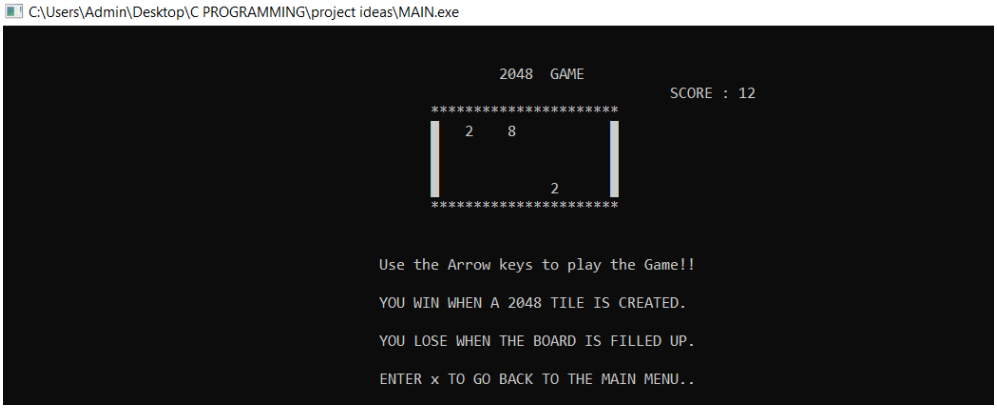
- Every time a move is done, a new tile is added to the board
- This is done using row-wise addition, and the numbers are moved to the right.
- And then a new number is added to the array – either 2 or 4.

- Now, let me click the up arrow,

- All the values in the board move upwards..

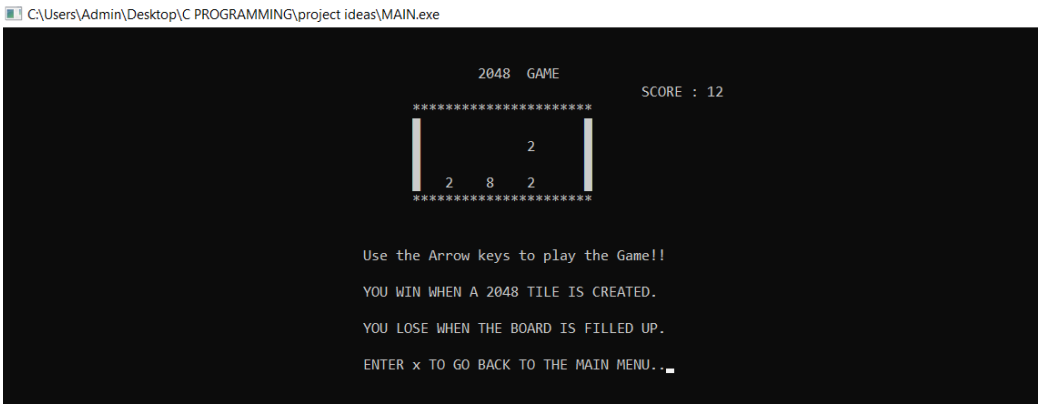
- This is done using **Bubble sort**, all the zeroes are swapped downwards.
- And a new tile 4 is created.

➢ Now, I will click the left arrow to add the 4's and create a tile with 8.



- Hence, eight is created.

➢ Now, I will press the down arrow



➢ Again down,

➢ Now, left..

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

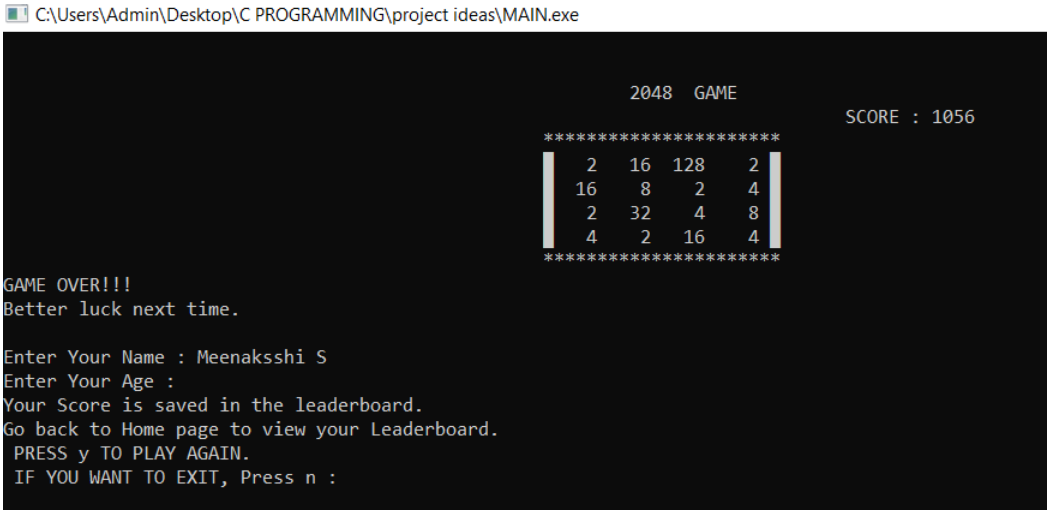
- Now, down arrow:

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

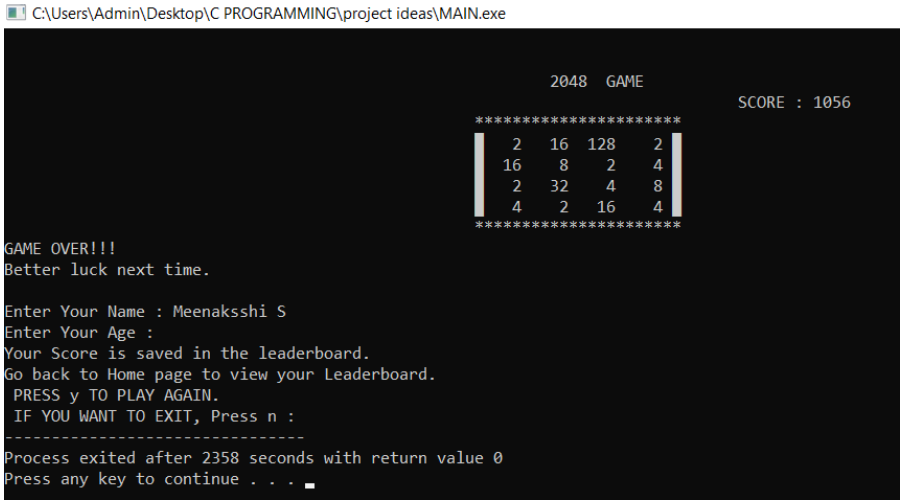
- 2's are added to get 4!!
- Hence, this is how the game proceeds.. I will show the output screen when the game is over.
- The score is calculated in the top right corner.. every time 2 tiles combine, the score increments the new value added.

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

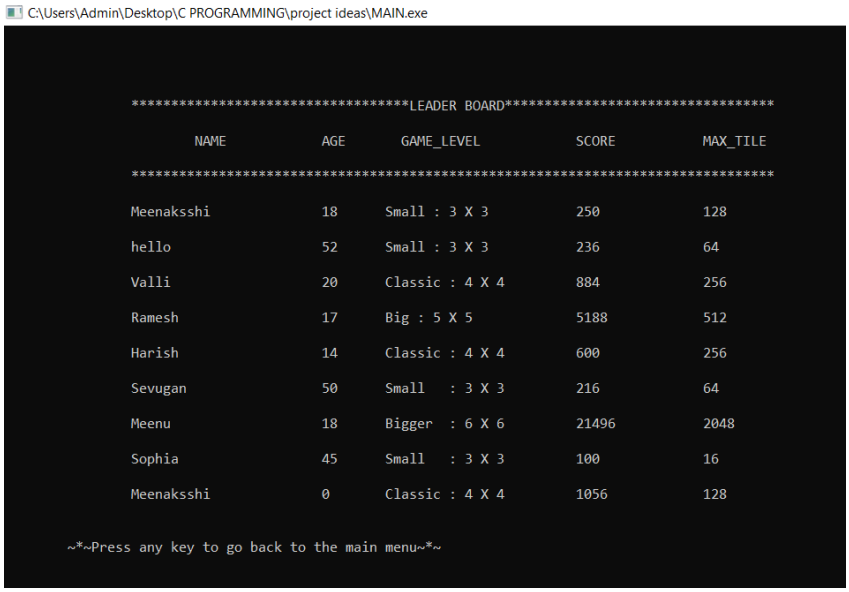
- Now, since all the tiles are filled.. and there are no empty tiles.. and also there are no two adjacent tiles having the same number.
- Hence the game is over and the player has lost the game since 2048 tile is not created.
- Now details are asked to store it in the leaderboard file.



- Now I will press n to exit.

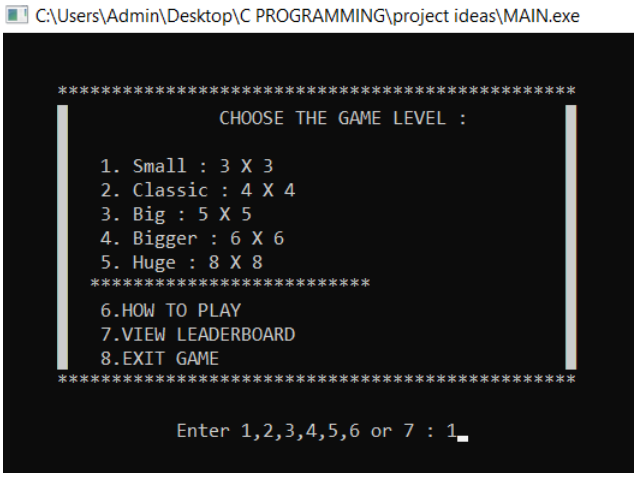


- **Hence my score is appended in the leaderboard FILE.**



Thus, all the other levels of the game operate in the similar way.

1. SMALL : 3 X 3 – EASY LEVEL



2. CLASSIC : 4 X 4 – BEGINNER LEVEL

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

```
*********************************************
|             CHOOSE THE GAME LEVEL :       |
|                                           |
|   1. Small : 3 X 3                        |
|   2. Classic : 4 X 4                      |
|   3. Big : 5 X 5                          |
|   4. Bigger : 6 X 6                       |
|   5. Huge : 8 X 8                         |
|  ************************                 |
|   6.HOW TO PLAY                           |
|   7.VIEW LEADERBOARD                      |
|   8.EXIT GAME                             |
*********************************************

          Enter 1,2,3,4,5,6 or 7 : 2
```

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

```
                    2048  GAME
                                        SCORE : 0
          **********************
          |             2      |
          |   2                |
          **********************

Use the Arrow keys to play the Game!!

YOU WIN WHEN A 2048 TILE IS CREATED.

YOU LOSE WHEN THE BOARD IS FILLED UP.

ENTER x TO GO BACK TO THE MAIN MENU..
```

3. BIG : 5 X 5 – MODERATE LEVEL

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

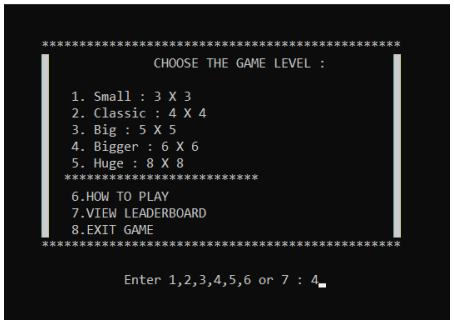
```
*********************************************
|             CHOOSE THE GAME LEVEL :       |
|                                           |
|   1. Small : 3 X 3                        |
|   2. Classic : 4 X 4                      |
|   3. Big : 5 X 5                          |
|   4. Bigger : 6 X 6                       |
|   5. Huge : 8 X 8                         |
|  ************************                 |
|   6.HOW TO PLAY                           |
|   7.VIEW LEADERBOARD                      |
|   8.EXIT GAME                             |
*********************************************

          Enter 1,2,3,4,5,6 or 7 : 3
```

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

```
                    2048  GAME
                                        SCORE : 0
          **************************
          |                        |
          |             2          |
          |        4               |
          **************************

Use the Arrow keys to play the Game!!

YOU WIN WHEN A 2048 TILE IS CREATED.

YOU LOSE WHEN THE BOARD IS FILLED UP.

ENTER x TO GO BACK TO THE MAIN MENU..
```

4. BIGGER : 6 X 6 – DIFFICULT LEVEL

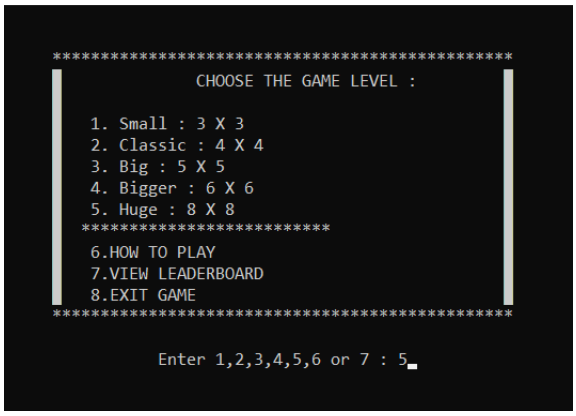
C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe



C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

5. HUGE : 8 X 8 – HARDEST LEVEL

C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

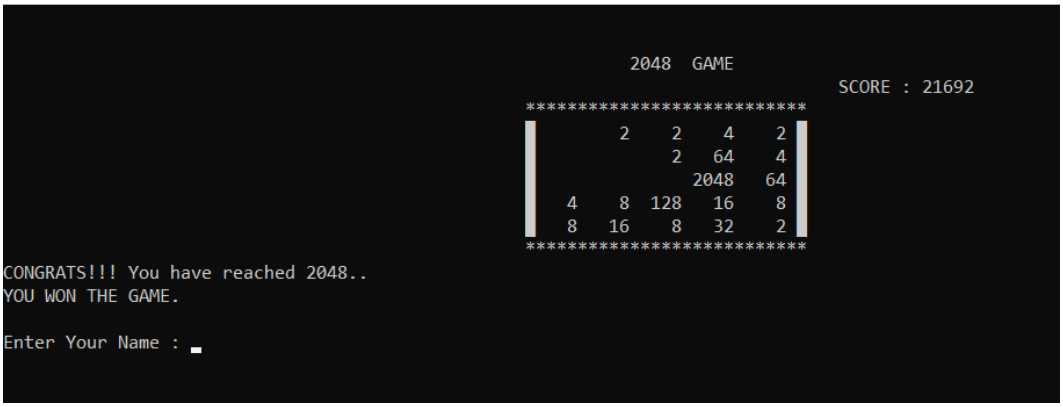


C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\MAIN.exe

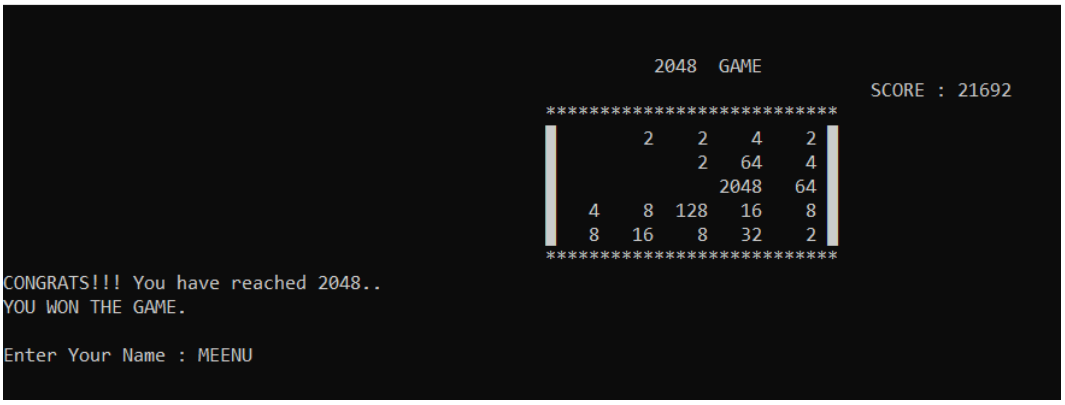
THE GAME THAT HAS BEEN WON:

- **2048 TILE HAS BEEN FOUND.. HENCE THE PLAYER HAS WON THE GAME.**

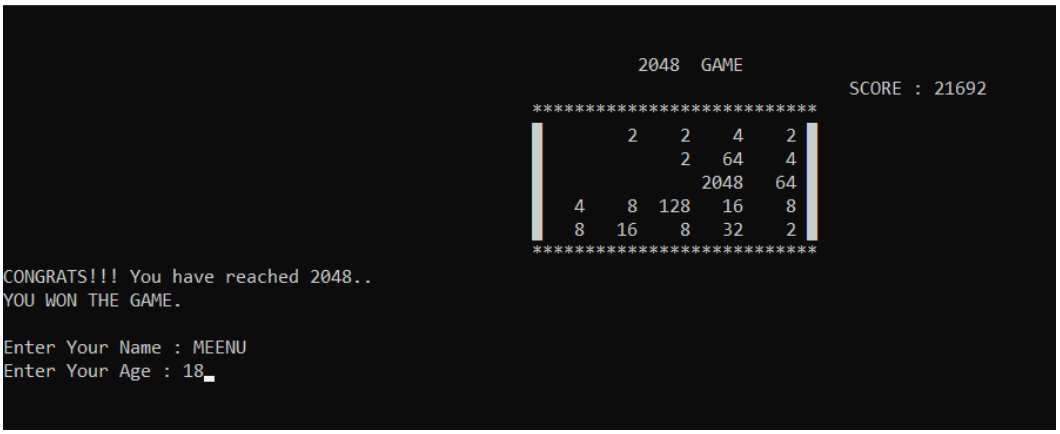
C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\2048 GAME\2048 MAIN GAME.exe



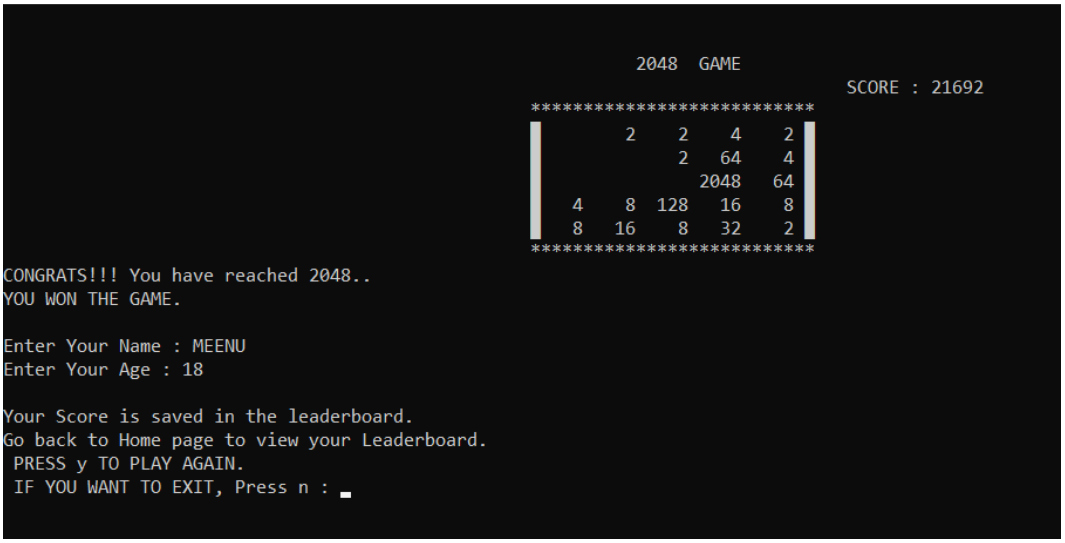
C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\2048 GAME\2048 MAIN GAME.exe



C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\2048 GAME\2048 MAIN GAME.exe



C:\Users\Admin\Desktop\C PROGRAMMING\project ideas\2048 GAME\2048 MAIN GAME.exe



MAIN C CONCEPTS USED IN THIS PROGRAM:

- 2D ARRAY
- STRUCTURES
- FILES
- POINTERS
- CONTROL STATEMENTS

FUNCTIONS USED AND THEIR ROLES:

- int randomno(int size);
  - generates a random value between 0 to (size_of_grid - 1)
  - It is used to choose a random empty tile on the game board
- int randomvalue();
  - 90% of the times 2 is added and only 10% of times 4 is added to the 2D array.. this is the rule of the game.
- void gotoxy(int x,int y);
  - It is used to move the cursor to the desired location on the screen.
- void display(int size, int tile[][size],int *ptrscore);
  - It displays the 2D array using nested for loops.
  - Instead of 0, a space is printed on the screen to avoid confusion during the game.
  - The score is printed in the top right corner of the screen
- int CheckEmpty(int size, int tile[][size]);
  - It checks the number of empty tiles in the game
  - It also searches the 2D array if 2048 number is present.
  - It returns:
    -1: the user has won
    0: the user has lost (ends the game)
    flag >=1:continue the game
- void Addno(int size, int tile[][size]);
  - It checks ifCheckEmpty is true
  - If yes, it adds a new number(2 or 4) to a random empty tile.
- void UP(int size, int tile[][size], int *ptrscore);
  - This function contains the code to add the numbers vertically
  - It also uses bubble sort algorithm to move the zeroes to the end in each column
  - It adds a new number only if there is any change in the array after the move.
- void LEFT(int size, int tile[][size],int *ptrscore);

- This function contains the code to add the numbers horizontally
- It also uses bubble sort algorithm to move the zeroes to the end in each row
- It adds a new number only if there is any change in the array after the move.

❖ void reverse_tile(char s[],int size, int tile[][size]);

- if “RIGHT” is passed to the function, then it flips the board horizontally
- else if “DOWN” is passed to the function, then it flips the board vertically

❖ int PlayGame(int size, int tile[][size],int *ptrscore);

- It displays a few basic rules of the game.
- It also accepts a char(arrow keys) and splits the code into parts using different switch cases
- After the arrow key is pressed, the respective function is called.

❖ void displaySCORE();

- The score is displayed.
- It opens the leaderboard file, extracts(reads) the data in the file and prints it accordingly.

❖ void Instructions();

- It contains a set of instructions if the player is new to the game.
- The game is explained thoroughly.

❖ void main();

- It displays the main page of the game which contains different levels
- The user is asked to choose the level
- Switch statement is used to store the size of the 2D array
- The max tile is calculated and stored in a structure variable.
- Appropriate messages are displayed if the player wins or loses
- Once the game gets over, the details of the player is asked and it is stored in the leaderboard file.