Candy Crush

Project for Comp 1011

Name: CHOUDHARY Muhammad Haris

Student ID: 18086809d

- Introduction

  The purpose of this project is to design a non-user friendly version of the game Candy Crush. The game is playable but does have any GUI to assist the user.

- Program Interface

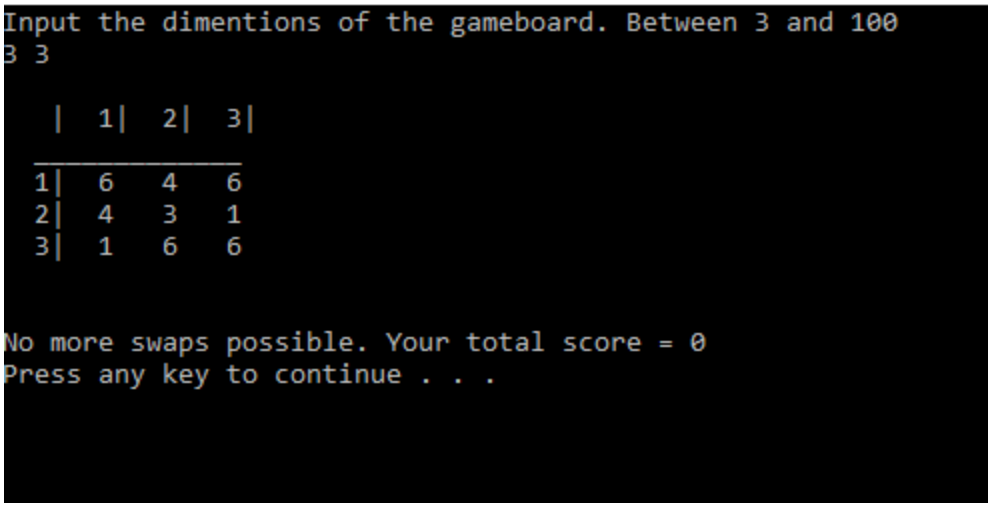
  The user can communicate with the program using keyboard. The program will run automatically as the user opens the Executable file. And the program does not have any preassigned termination command, but the program terminates automatically when there are no possible swaps possible.

- Program Execution

  1. The algorithm begins with taking the dimentions of the gameboard, an array of integers, from the user.
  2. Then a gameboard is randomly generated by the program and is displayed on the screen. Each candy are displayed by a unique number between 1 and 6.
  3. The program checks all the candies for any more than 2 similar candies in a row or a column. And each similar candy is replaced by -1 in the array and a score for each of the replaced candy is added to the players score.
  4. All the -1s in the array are replaced by the numbers above it. And a random number is stored at the top.
  5. When there are no more consecutive similar candies left the program asks the user for inputing the indexes of the candies to be swapped. And the user is given the hint for the candies that could be swapped.
  6. The user inputs a value and the candies are swapped in the array.
  7. The program repeats all the steps from 3-6 until there no more possible swaps left. And the game is ended with the users score printed.

```
Input the dimentions of the gameboard. Between 3 and 100
10 10

   |  1|  2|  3|  4|  5|  6|  7|  8|  9| 10|
 ____________________________________________
  1|  6   4   5   3   1   6   5   1   3   5
  2|  3   3   5   5   2   2   6   5   6   6
  3|  1   5   4   4   6   4   3   4   6   1
  4|  5   4   1   2   4   6   3   5   3   6
  5|  1   1   6   3   6   6   4   5   2   1
  6|  2   6   4   6   4   2   2   5   1   1
  7|  2   1   5   4   1   2   2   6   6   5
  8|  4   6   3   1   5   1   1   6   5   3
  9|  4   1   4   5   5   6   2   2   1   6
 10|  4   5   2   2   5   2   4   6   2   2



   |  1|  2|  3|  4|  5|  6|  7|  8|  9| 10|
 ____________________________________________
  1|  6   4   5   3   1   6   5   1   3   5
  2|  3   3   5   5   2   2   6   5   6   6
  3|  1   5   4   4   6   4   3   4   6   1
  4|  5   4   1   2   4   6   3  -1   3   6
  5|  1   1   6   3   6   6   4  -1   2   1
  6|  2   6   4   6   4   2   2  -1   1   1
  7|  2   1   5   4   1   2   2   6   6   5
  8| -1   6   3   1  -1   1   1   6   5   3
  9| -1   1   4   5  -1   6   2   2   1   6
 10| -1   5   2   2  -1   2   4   6   2   2
```

```
   |  1|  2|  3|  4|  5|  6|  7|  8|  9| 10|
 ____________________________________________
  1|  1   4   6   4   5   6   5   5   3   5
  2|  6   3   5   3   3   2   6   4   6   6
  3|  3   5   4   4   1   4   3   6   6   1
  4|  6   4   1   2   1   6   3   1   3   6
  5|  3   1   6   3   2   6   4   5   2   1
  6|  1   6   4   6   6   2   2   4   1   1
  7|  5   1   5   4   4   2   2   6   6   5
  8|  1   6   3   1   6   1   1   6   5   3
  9|  2   1   4   5   4   6   2   2   1   6
 10|  2   5   2   2   1   2   4   6   2   2

Score: 12
swap possible at index 1 , 5 and 1 , 6
Input the index of candies to be swapped
1 5
1 6

   |  1|  2|  3|  4|  5|  6|  7|  8|  9| 10|
 ____________________________________________
  1|  1   4   6   4   6   5   5   5   3   5
  2|  6   3   5   3   3   2   6   4   6   6
  3|  3   5   4   4   1   4   3   6   6   1
  4|  6   4   1   2   1   6   3   1   3   6
  5|  3   1   6   3   2   6   4   5   2   1
  6|  1   6   4   6   6   2   2   4   1   1
  7|  5   1   5   4   4   2   2   6   6   5
  8|  1   6   3   1   6   1   1   6   5   3
  9|  2   1   4   5   4   6   2   2   1   6
 10|  2   5   2   2   1   2   4   6   2   2
```



- Input and Output

  - ✓ The inputs are the dimentions of the game board and the indexes to be swapped.
  - ✓ The outputs include the array (game board) that is printed after any changes to the array, the score of the player, hints for swapping, a few prompts for input and error messages for invalid input.

- Program Structure

  The program comprises of 8 functions as follows

  - ✓ *void genGameBoard(int gameboard[N][N], int x, int y);*

    This function generates the gameboard by using random integer values according to the dimensions input by the user .

  - ✓ *void PrintGameBoard(int gameboard[N][N], int x, int y);*

    This function prints the gameboard.

  - ✓ *void fillUp(int gameboard[N][N], int x, int y);*

    This function brings down the candies on top of the ones replaced by -1.

  - ✓ *bool checkSimilar(int gameboard[N][N], int x, int y);*

    The function checks for the consecutive similar candies in the array. And replaces them with -1

  - ✓ *void swap(int gameboard[N][N], int i, int j, int k, int l);*

    This function swaps the candies suggested by the player.

  - ✓ *void scoreCount(int gameboard[N][N], int x, int y, int *score);*

    This function counts the score of the user.

- ✓ *bool hint(int gameboard[N][N], int x, int y);*

  This function generates the hint of the possible swaps for the player. It calls "*hintchecker*" to check for the possible swaps.

- ✓ *bool hintchecker(int gameboard[N][N], int x, int y);*

  This function checks for the consecutive similar candies to generate the hint for the user.

- Difficulties Encountered

  There were a few problems with colour coding the array as there were no libraries available for the colours.