Corner Cases :

In this game we have a lot of corner cases , and in this file we will try to discuss it and show how we solve these corner cases .

1. **if the input is character instead of number**

   The first problem we might had is that the user has entered a character instead of a number Like in the **start menu** when the program asked the user to enter a number from 0 to 4 **or** when the user is asked to enter the number of the column in the game .
   We solved this program by using the two built-in functions **isalpha()** and **atoi()** , isalpha function which returns 1 if the argument is alphabetic character and 0 if the argument is not alphabetic (number) but it takes the argument of data type **char** only so we take the input from the user as an array of characters which called **buffer** of length 20 And then asked the isalpha function to detect whether the input is alphabetic or not in a while loop , if it's alphabetic then we will ask the user to enter a number by sending the message " not possible , please enter a number " , And then when the user enters a number (When the isalpha function returns 0) we will convert the data type from **char** to **int** by using the second built-in function **atoi** which takes the **character** then converts it into an **integer** data type .

```
while (isalpha (buffer[0]) != 0 )
{
    printf("\n not valid , please enter a number ");
    scanf("%s", buffer);
}
theColumn = atoi (buffer) ;
```

2. **if input is invalid**

The second corner case is when the user enters an invalid input Like **Y** or **N** in the end of the game or not entering a number from 0 to 4 in the start menu (e.g. entering a number more than 4 or less than 0) .

We will solve the first problem (**Y** or **N**) by using an if – else if – else statement , if the user enters **N** the program will show a message " Have a good day " ,

And if the user enters **Y** the program will call the function **start game ( )** to play again , and if the user enters anything except **Y** or **N** the program will show a message " Not a valid option. please choose Y or N ! " Then it goes to a Label (in this example it's called Ramy) by using goto instruction to repeat the code again and tests whether the user enters a **Y** or **N** .

```
    Ramy :
    scanf(" %c", &playAgainToken); // Get the user input if the player wants to play again.

if(playAgainToken  == 'N') {
    printf("\n Have a good day");
    exit(1);
}

else if(playAgainToken  == 'Y') {
    startGame();
}
else
{
    printf("\n Not a valid option. please choose Y or N ! ");
    goto Ramy ;
}
```

And we will solve the second problem of entering an input which is out of range by the same previous code And in case that the user enters a number which is more than 4 or less than 0 the program will show a message " not valid , please enter a number from 0 to 4 " .

3. if the two players enter the same name

In this case if the two users write the same name when the program asks them to enter it in the beginning of the game , And we consider it a problem as if the two players pause the game for a while and then return back to it then some mess might happens , So we if the second player enters the same name of the first player The program will ask the user to change it by showing a message

" Sorry Player O , You Entered The Same Name of player X , Please change it ! "

And we done this by using the built-in function **Strcmp ( )** and an if condition .

```
if(strcmp(playerOneName,playerTwoName) == 0) { // If the players names are identical for the name
    printf("\n Sorry Player O , You Entered The Same Name of player X , Please change it ! ");

    printf("\n Player O: Enter your name please."); // Asks player 2 to enter
scanf("%s", playerTwoName);
```

4. if the entered column is full

In this case we will consider that the user has entered a number of column which is a full column , The Program will show a message " Not possible , This Column Is FULL !! " .

We have done this by using an array of integers of size 7 (One For Each Column) which is called **max_Horizon** and we gave him an initial value of { 0 } ,

In every time any of the two users chooses a specific column , The index of the array which is the same number of the column will be incremented by 1 until it reaches its maximum value which is considered to be 6 (As the game has 6 rows) then if the user tries to enter the number of this column (which is full) again it will shows him the message which is mentioned before and then asks the same user to enter a valid column which is not full .

```
while ( max_Horizon [theColumn]>=7 )
{
 printf("\n Not possible , This Column Is FULL !! ");
  valid_move = 0; // Move is not valid
   if(counter % 2 == 0)
   {printf("\n Player %s it's your turn now. Hit 0 to Save Game ", playerTwoName);
  scanf("%d", &theColumn);
   }
   else
   {
       printf("\n Player %s it's your turn now. Hit 0 to Save Game ", playerOneName); // Prompts the player to make a move.
  scanf("%d", &theColumn);
   }

}
```

5. if not valid column

In this last case we will discuss if the user enters an out of range value of column which is must be from 1 to 7 .We will use a while statement to test whether the input more than 7 or less than 0 (As we use 0 for saving the game) Then it will show him a message " Not possible , Please Enter A Number From 1 to 7 !! " .

```
while(theColumn > BOARD_VERTICAL + 1 || theColumn < 0 )
{ // If the column that the user
    printf("\n Not possible , Please Enter A Number From 1 to 7 !! ");
```