VRIJE
UNIVERSITEIT
BRUSSEL

Project Computersystemen

MINESWEEPER

Group G06

Abdessamad Nassihi& Mohamed Azahaf

December 25, 2022

Academic year 2022-2023
Industriële wetenschappen

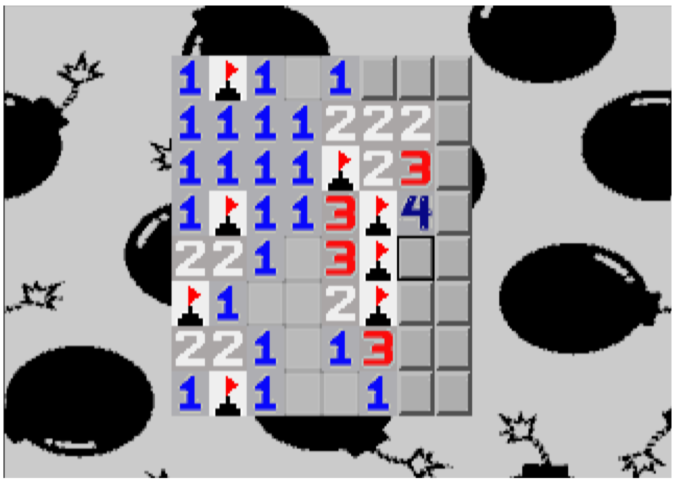
Figure 1: Minesweeper

1 Introduction

The project of this course consists of making a game on the 80386 32-bit protected mode assembly language.For compiling the code we use the Borland Turbo Assembler(TASM) wich is running inside the DOSBox environment.

The game we decided to make is called minesweeper. The main goal of this game is to find all the mines present in the field. The field is made of 64 squares each containing a number or a mine. To see the content of the square, the user will have to press a specefic key. The user can place a flag on a square if he thinks that it contains a mine. If a square contains a number,this number indicates how many mines there are in his surrounding.

2 Manual

After compiling the code, a menu will first appear on the screen where the commands of the game are explained:

- The arrows for moving the cursor
- Space for unveiling the square where the cursor is pointing to
- The letter F for placing or removing a flag

For starting the game the user will have to press the ENTER key, a new screen will then appear where the user will be able to choose between the different modes:

- Easy which contains 5 mines
- Normal which conatins 10 mines
- Hard which contains 15 mines

Once a mode has been chosen, the field will appear on the screen and the user can start playing. A GAME OVER screen will appear if the user has unveiled a square containing a mine. In the same manner a YOU WIN! screen will appear at the moment the user has found all the mines. The number of flags the user can place corresponds to the number of mines present in the field. The user can terminate the program at any moment by pressing the ESCAPE key.

3 Program features

The game has the following features:

- The mines are placed randomly in the field
- The first move will always be safe
- Flags can be used to indicate where a mine is
- The user cannot place more flags than the amount of mines in the field
- The game has three different modes that defines the difficulty
- Right before the 'GAME OVER' screen, the positions of the other mines are shown

4 Program design

The sprites of the game are generated from images converted to bin files. Each sprite has its own color on the palette determined by the given offset.

The array 'Field' indicates where a mine is present in the field. The elements in this array have either the value 0(not a mine) or 1(a mine).The array 'Squares' indicates if a specefic square has been unveiled. At the start all the elements have te value -1, which means that the square has not been unveiled. After revealing the square the value at corresponding position in the array will get a value between 0 and 7. The value between 0 and 5 represents the number of mines in his surrounding, 6 represents a mine and 7 a flag.

When moving the cursor, its position gets updated and the square where the cursor came from is drawn again. This will make impression that the cursor is moving.

In order to place the mines randomly, values between 0 and 1 will be generated for the array 'SelectRow' and values between 0 and 7 will be generated for the array 'PosOfMine'. The array 'SelectRow' indicates how many mines are present in each row and 'PosOfMine' indicates the position of the mines in each row.

Figure 2 shows the flowchart of the code

5 Encountered problems

Overall, the project went smoothly. The main challenge for us was displaying the sprites but after some research we managed to do it. Another aspect that was difficult, was making the field as random as possible.

6 Conclusion

The game is as good as finished in the way that it has the most important features of a minesweeper game. To make it even more complete we could add a feature that makes the game playable with a mouse.

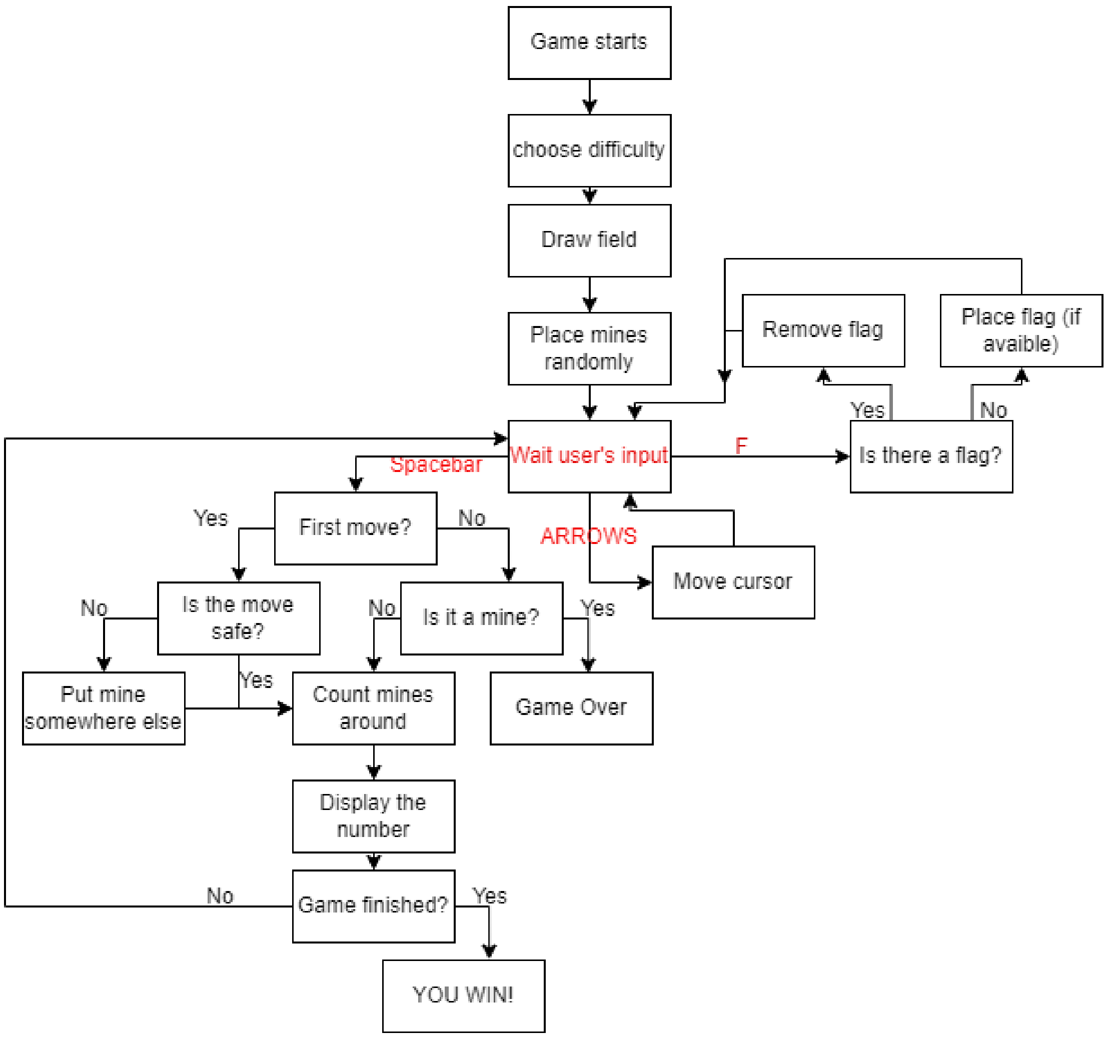

Figure 2: Flowchart