|  |  |
| --- | --- |
| **Mandatory Case** | Description: LogoBINUS-University |
| C |
| **Periode Berlaku** Semester Genap 2020/2021  ***Valid on*** *Even Year 2020/2021* | **Software Laboratory Center**  **Assistant Recruitment 21-2** |

## Soal

*Case*

**A Bomber**

You were cleaning the attic and then you saw your old game console, the Famicom. When you were little, you used to play the famous game Bomberman in Famicom the whole time (<https://youtu.be/3smytj9Bu_E>). You are now swimming in memory lane and then **you thought it would be a great idea to recreate this game in C programming language** as you are trying to hone your coding skills. You knew that it would be too hard to recreate the game with the same feature so you decided that you will only create the game with these features :

* **Prerequisite**
  + You have to **read the file “map.txt” to load the map** for your A Bomber game.
  + **Read the file “save.txt” to load the save data** which **contains the highscore and player name**. The format is as following :

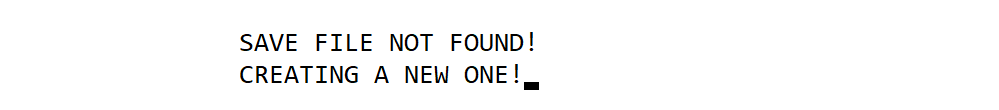
NAME#SCORE  
NAME#SCORE

Figure 1. Save File format.

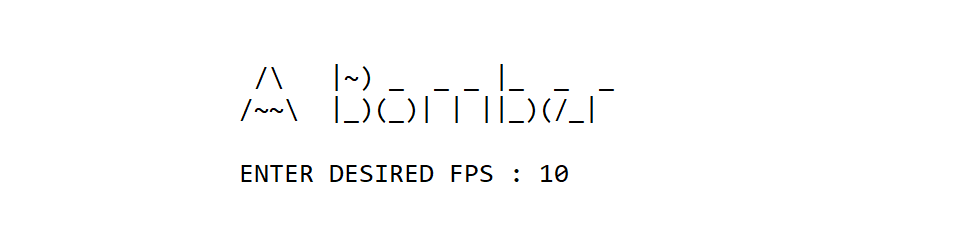
* + **If the “map.txt” file is not found** then show an **error message** and **exit the game**

 Figure 2. Map not found message.

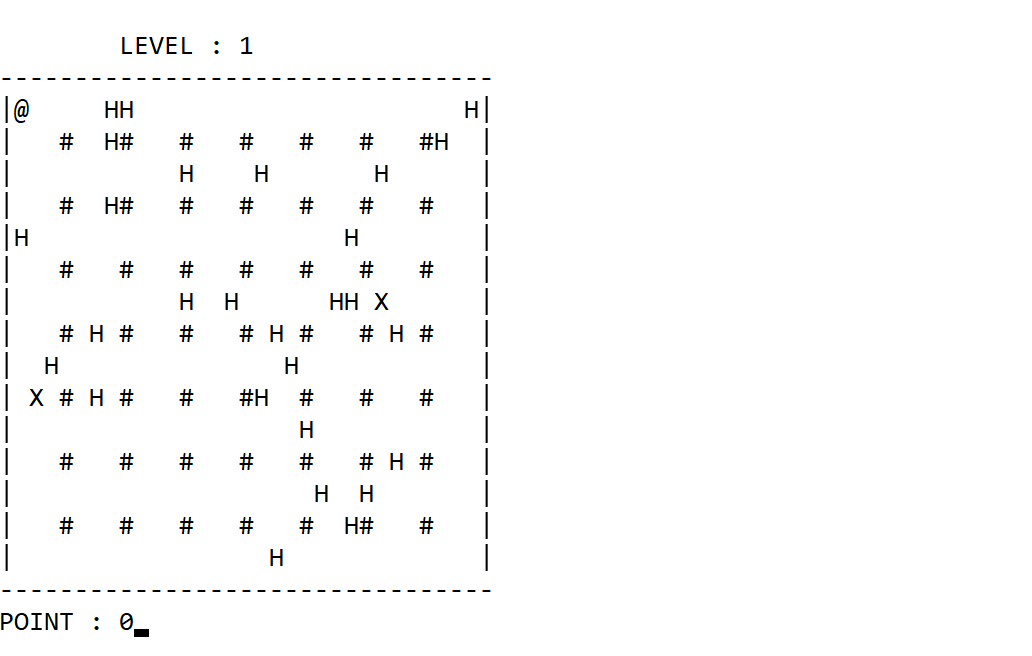
* + **If the “save.txt” file is not found** then show an **error message** and **create “save.txt” file**

 Figure 3. Save file not found message.

* + **You are not required to create the FPS feature as provided in the example program** since it’s just for ease of use (aesthetic)’s sake, but if you want to then go ahead.

 Figure 4. FPS input feature.

* **Gameplay**
* When the user starts the program it **automatically runs**.

 Figure 5. Start game.

* + 1. **Generate a destructible wall ( H ) randomly**, with the following specifications :
       - The number of destructible wall must adhere to this specific formula where ***n*** is a random number between 9 – 18 inclusively:

Figure 6. Wall count formula.

* + - * Make sure that the generated **destructible wall is placed in an empty space** (not the outer wall nor the indestructible wall ( # ))
    1. **Generate enemy**(s)
       - Enemy ( X ) must be **placed randomly within the outer wall** and **in an empty space.**
       - The **enemy count for level one is two** and **increments every level up.**
* **The player ( @ ) will be placed at point (1,1)** in the map **at the beginning of each level**.
* Player can **move using WASD**
* Player **can only move to an empty space**
* Player **can place a bomb ( \* ) by pressing SPACE BAR**. The **bomb will be placed according to the player’s last move direction**.
* Player can **go to the next level by going to a door (O)** that is **hidden underneath a destructible wall (H).**

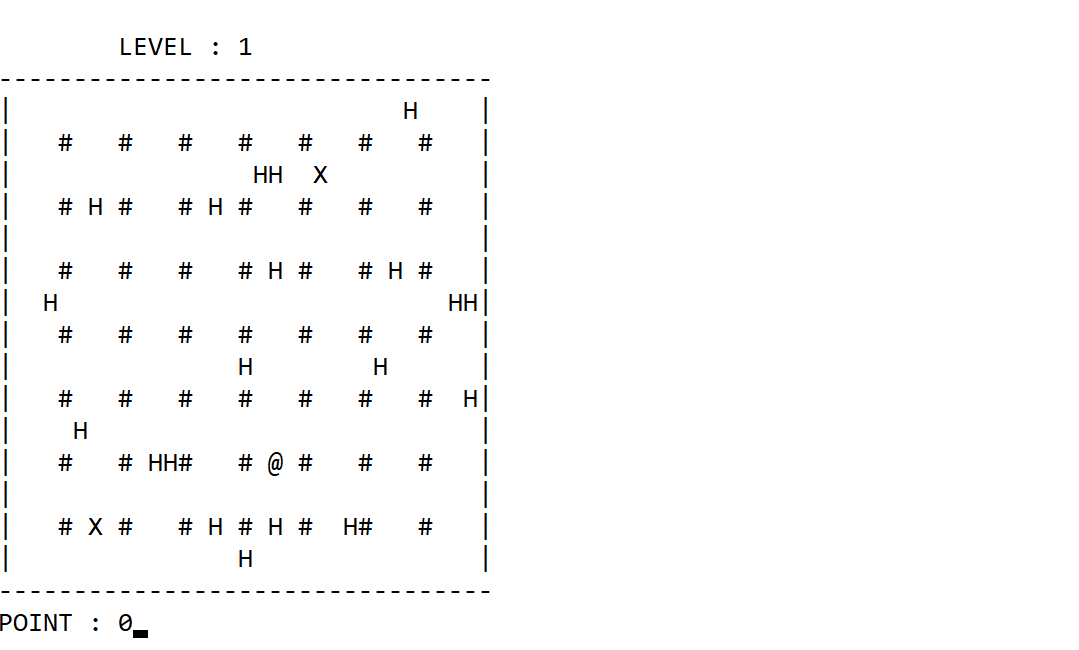


Figure 7. Player about to place a bomb in his down direction.

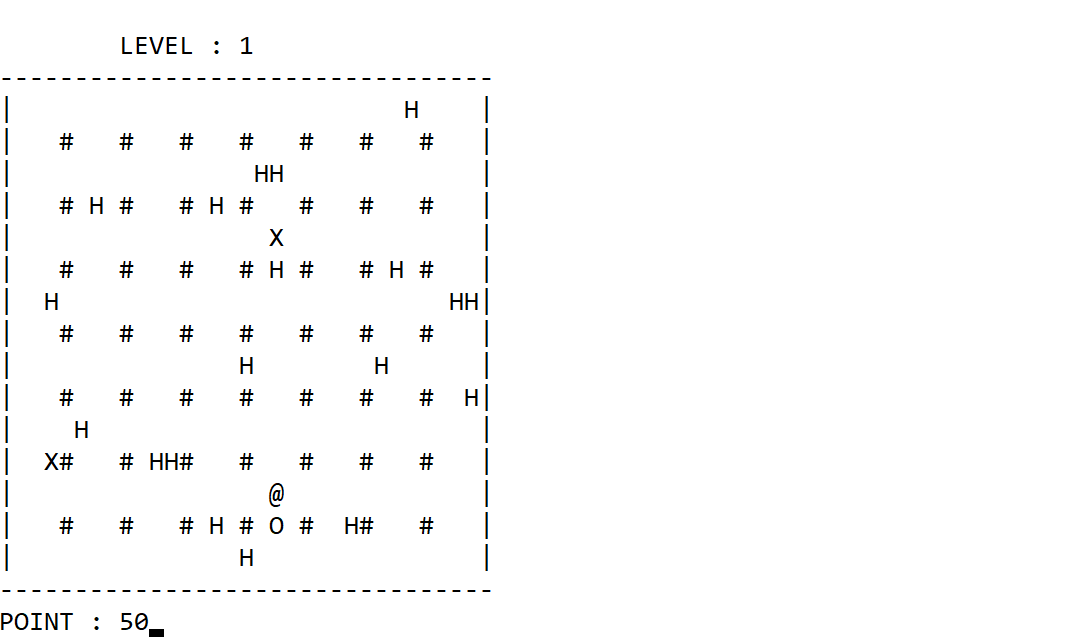


Figure 8. The destroyed wall revealed a door to the next level

* The **bomb will detonate after planted for 2 seconds** and the **detonation area** is as following :

OOO

O\*O

OOO

O is the space that will be destroyed and \* is the bomb

Figure 9. Detonation Area.

1. **Please note that the bomb cannot destroy the outer wall** ( - ) and ( | ) **nor the indestructible wall** ( # )
2. **Player can place as many bombs as he wants** and **it will detonate according to its planted time**.
3. **If the bomb destroyed a destructible wall ( H )**,
   * **The wall will be removed** from the map
   * The player will gain points according to this formula for each wall destroyed:

Figure 10. Points to be gained for each wall destroyed

1. **If the bomb destroyed an enemy ( X ):**
   * **The enemy will not be active** anymore
   * **The enemy will be removed** from the map
   * The player will gain points according to this formula for each enemy destroyed:

Figure 11. Points to be gained for each enemy destroyed

1. **If the bomb destroyed the player**, **the player lost**. Show the **losing scene** and the **user will be prompted to save** the score.

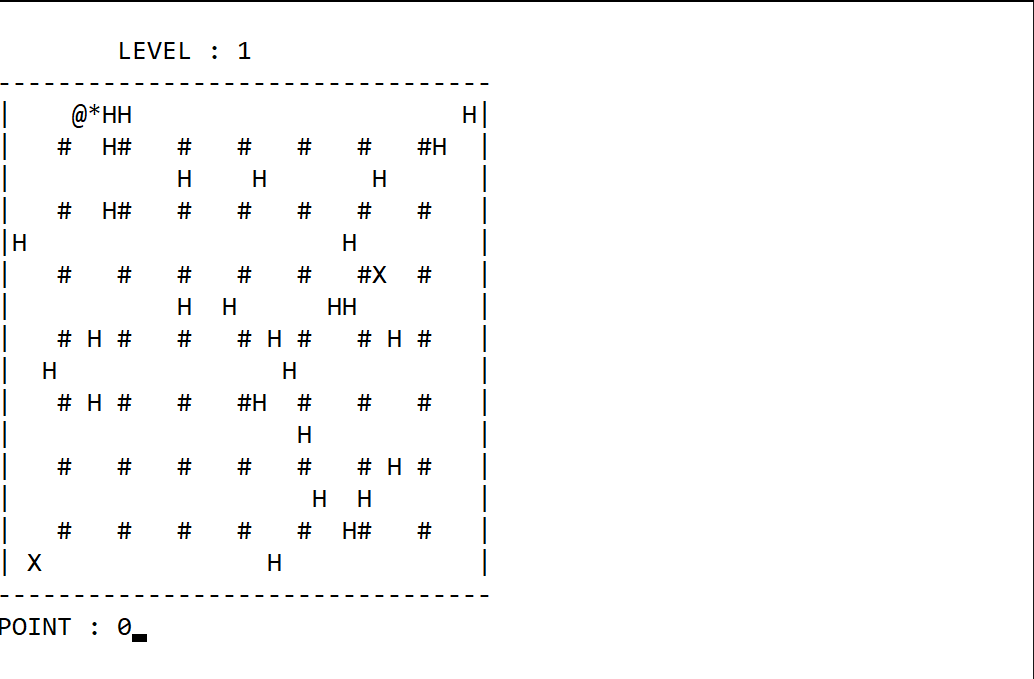
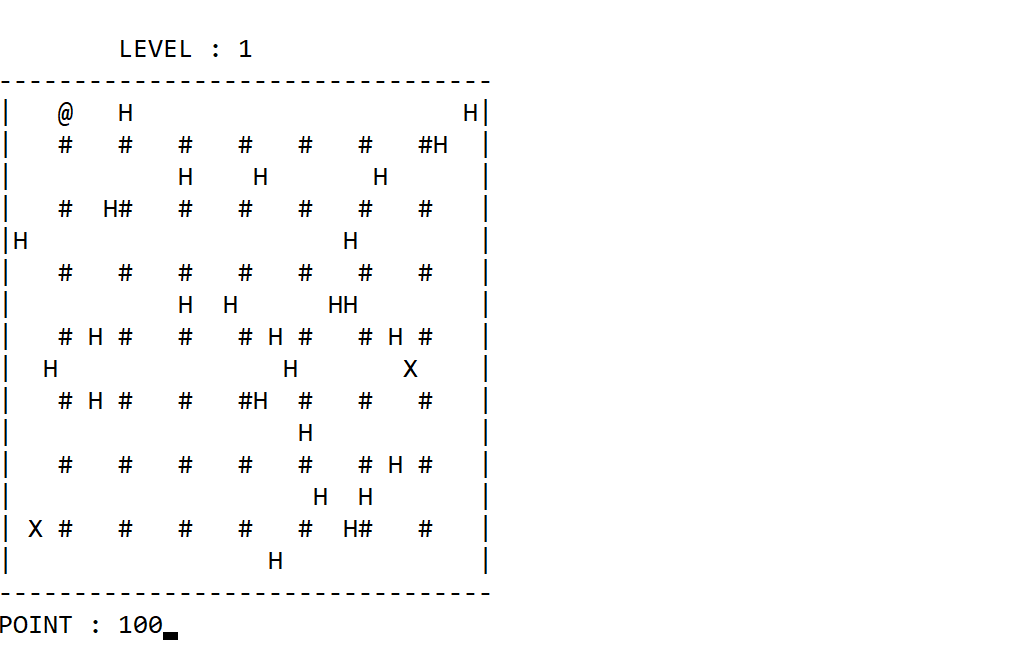
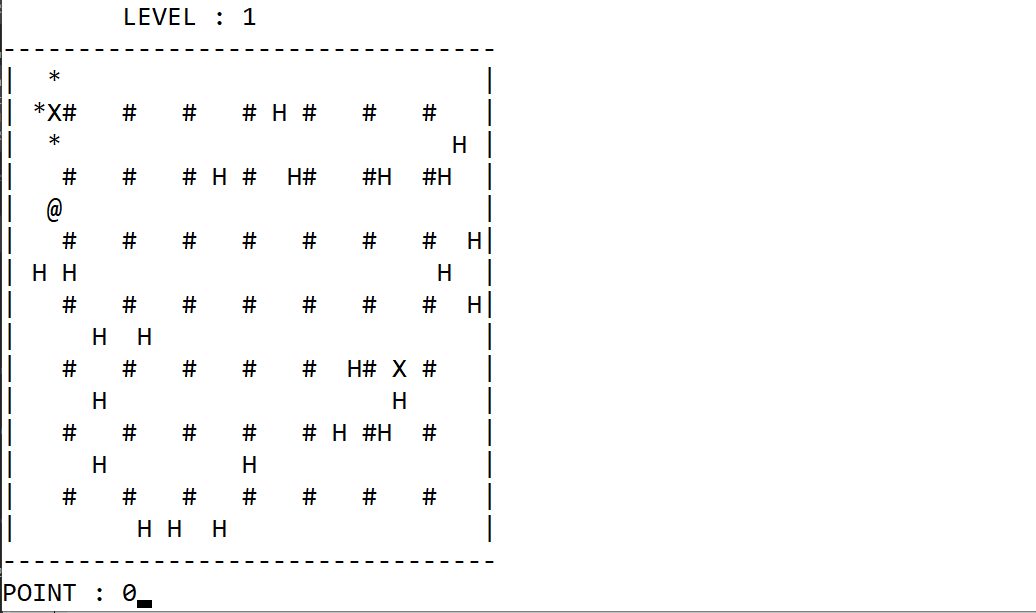
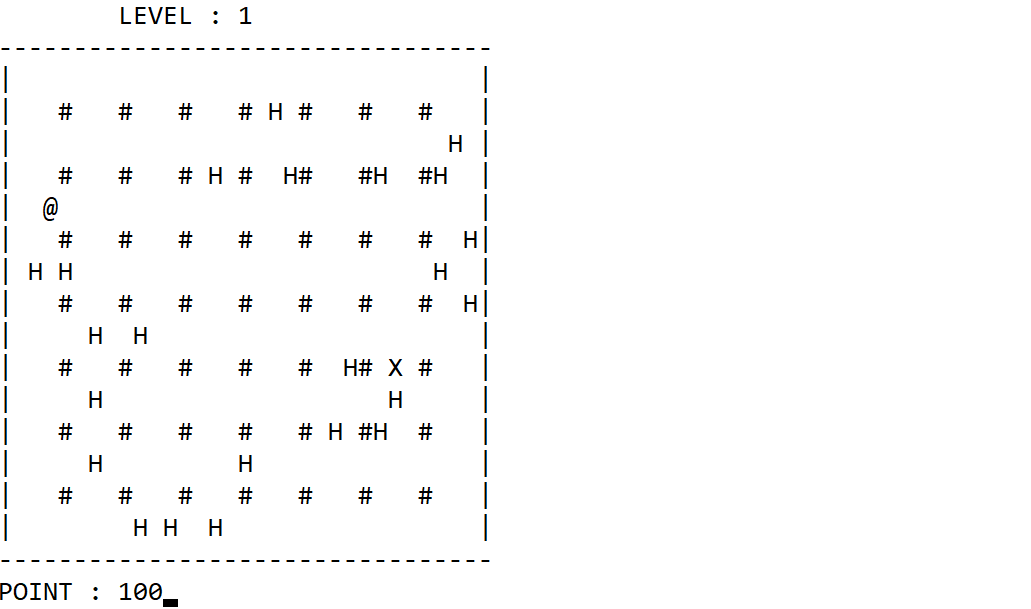


Figure 12. Player planted a bomb in the upper left part of the map near walls.

 Figure 13. The bomb detonated an area (2 walls) and player gained 100 points (50 x 2).

 Figure 14. Player planted a bomb in the upper left part of the map near the enemy.

 Figure 15. The enemy is removed and the player gained 100 points for one enemy.

* The enemy(s) will **move randomly either up, bottom, left, or right**.

1. Validate that enemy **can only move to an empty space**
2. Validate that enemy only **move once every second**
3. If there is **no empty space available** to move then the **enemy will not move**
4. If the **enemy moves to where the player is**, the **enemy eats the player** and the **game is over**. Show the **losing scene** and the user will be **prompted to save** the score.

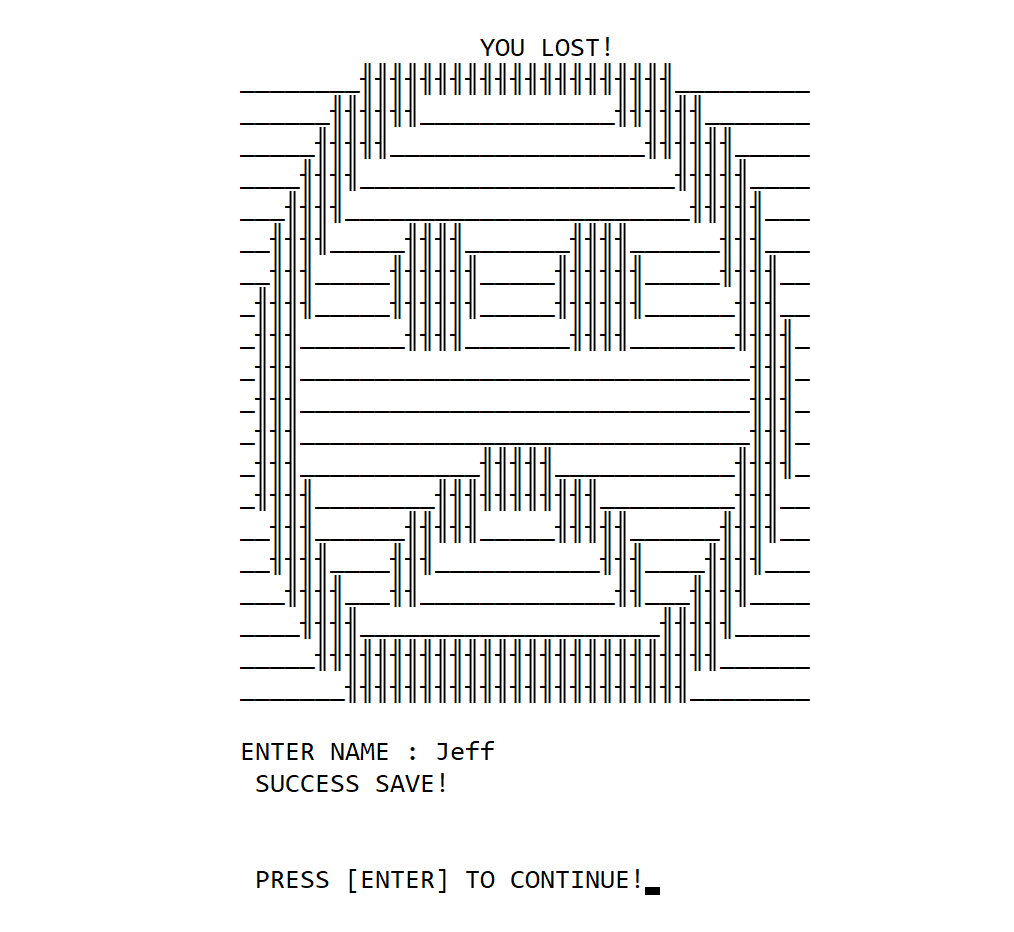


Figure 16. The player lost (either exploded by a bomb or eaten by the enemy).

* **Menu**
  + While playing, the **user can press ESC key to pause the game** and see the menu.

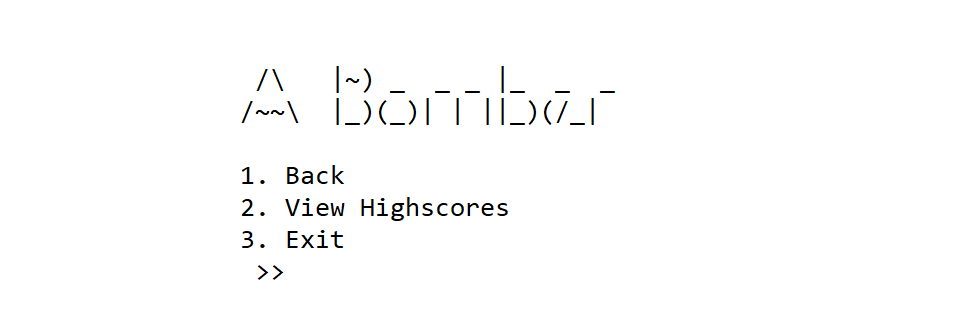


Figure 17. Menu.

* + **To go back** to the game user can **choose menu number one (1)**. Note that the **previous state of the game must be preserved**. Nor enemy nor the wall may be regenerated!
  + Menu number two (2) allows the user to **view all players’ name and their high scores from the “save.txt” file.** 
    1. Please keep in mind that the **aesthetics of your program will not be scored**.
    2. Then the player can **press ENTER key to return to the game**.

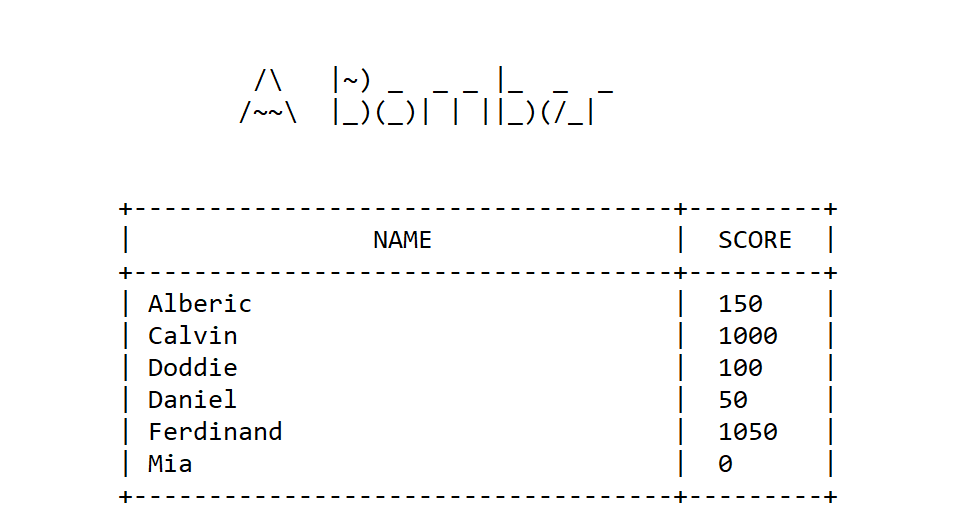


Figure 18. View highscore.

* + Menu number three (3) **prompts the user to save his current score and name** (see Save game point), **then exit the program**.
* **Save game**
  + Prompt the user to **input name.**
  + Validate that **name cannot be empty.**
  + Validate that **name must consist of minimum two character**.
  + Validate that **name must only contain one word**.
  + **Save the inputted player name and the current score (point) to file “save.txt”** with the specified format as described in figure 1.

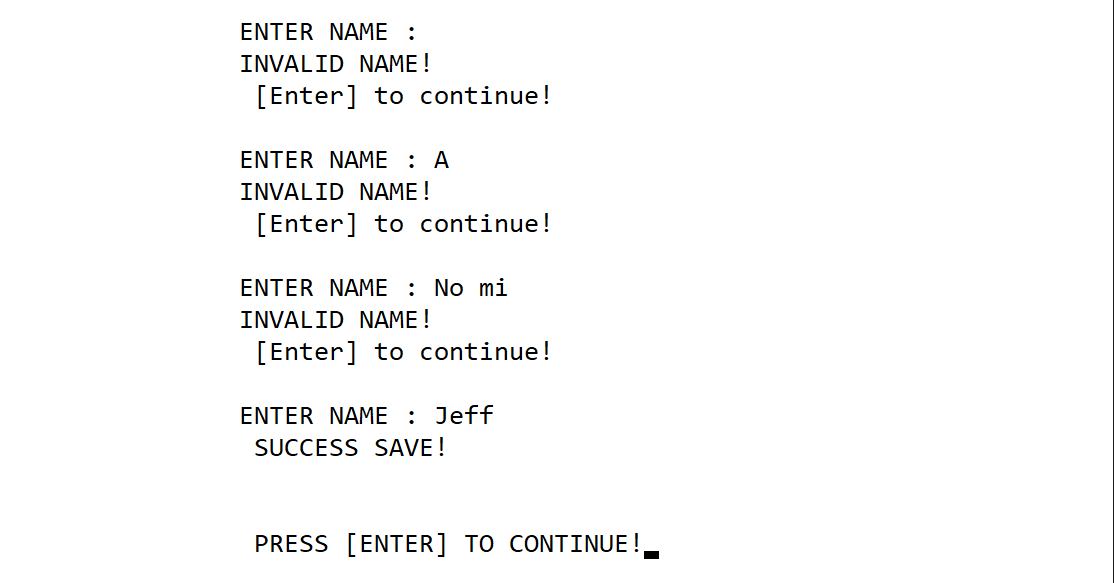


Figure 19. Save game.

Please run the EXE file to see the sample program.