COMPUTER GAMES

FINAL PROJECT: JUMPMAN

TEAM MEMBERS:
REFİYE ŞEHNAZ YILDIRIM – 171805012
KANİ SUNGUR – 161805049

13/06/2021

CONTENTS

1. HISTORY AND DEVELOPMENT OF JUMPMAN

Jumpman is a platform game written by Randy Glover and published by Epyx in 1983.

Jumpman came about after Glover saw Donkey Kong in a local Pizza Hut. This led him to become interested in making a version for home computers.

The initial version was written using a compiler on the Apple II, moving the software to the Atari. A prototype with 13 levels took four or five months to complete. After looking in the back of a computer magazine for publisher, in early 1983 he approached Broderbund. They were interested but demanded that their programmers be allowed to work on it. The next day he met with Automated Simulations, who were much more excited by the game and agreed to allow Glover to complete it himself.

At the time, the company was in the process of moving from the strategy game market to action titles, which they released under their Epyx brand. Jumpman was the perfect title for the brand, and the company hired him. Aiming the game at the newly enlarged RAM available on the Atari 800 led to the 32 levels of the final design. The Atari release was a huge hit, and the company soon abandoned their strategic games and renamed as Epyx. Glover then moved on to a C64 port, which was not trivial due to a particular feature of the Atari hardware Glover used to ease development.

The C64 port has an improved Jumpman sprite, but is otherwise similar to the Atari version. Other programmers at Epyx ported it to the Apple II, with poor results, and a year later, contracted Mirror Images Software for an IBM PC/PCjr port. The Atari and Commodore versions were released on disk and cassette tape, the Apple and IBM versions only on disk.

After developing the original versions, Glover moved on to Jumpman Junior, a cartridge title with only 12 levels. He stated that it wasn't really a sequel to Jumpman, but more of a "lite" version for Atari and Commodore users who didn't have disk drives. These versions removed the more complex levels and any code needed to run them. The C64 version was later ported to the ColecoVision, which used the C64 levels. [1]

2. GAMEPLAY

According to the story, the base on Jupiter has been sabotaged by terrorists who have placed bombs throughout the base's three buildings. The object of the game is to defuse all the bombs in a platform-filled screen. Jumpman defuses a bomb by touching it. Jumpman can jump, climb up and down ladders, and there are two kinds of rope each allowing a single direction of climbing only.

The game map is organized into a series of levels, representing the floors in three buildings. When all of the bombs on a level have been deactivated, the map scrolls vertically to show another floor of the building. When all of the levels in a building are complete, a screen shows the remaining buildings and moves onto the next one. The order of the maps is randomized so players do not end up trapped on a level they cannot complete.

Hazards include falling smart darts, fall damage, and other hazards that are unique to a certain level. Upon being hit or falling from a height, Jumpman tumbles down to the bottom of the screen.

Points are awarded for each bomb defused, with bonus points available for completing a level quickly. Jumpman's game run-speed can be chosen by the player, with faster speeds being riskier but providing greater opportunity to earn bonus points. [1]

2.1 Game Features

There are 6 options on the main display screen of the game. These:

- Beginner
- Intermediate
- Advanced
- Grand Loop
- Randomizer
- Practice

After making the first selection, it's our turn to choose the player. There is a choice between 1-4.

After the game starts, ladders, ropes, coins, enemy and player appear.

The enemy can move along the x and y axis. It can change direction abruptly depending on the player's position.

The player can go right and left along the platforms. The player can move up and down ladders. The player cannot cross the border of the ladder.

If player goes too close to the far right and left on the platform, player will fall and lose life in the game. The player has 6 lives per level.

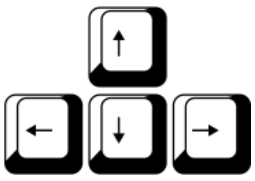
If player can pass the level without losing all his/her lives, then the new playing field will appear by sliding down from above.

The player is provided with an area to practice.

By practicing in this area, you can perform better in the main game and achieve a higher score.

A certain point is collected for each money collected in the game. As the levels get harder, the points of the collected money increase.

2.2 Game controls



Left

Right

Up

Down

Jump

1-8: Set the game speed, press key before the start of each level while the current player is displayed. 1 is fast, 4 is default and 8 is slow. Arrow keys are used to advance the player. It uses the space key for the player to jump. The player can move left and right on the platform and up and down on the ladder.

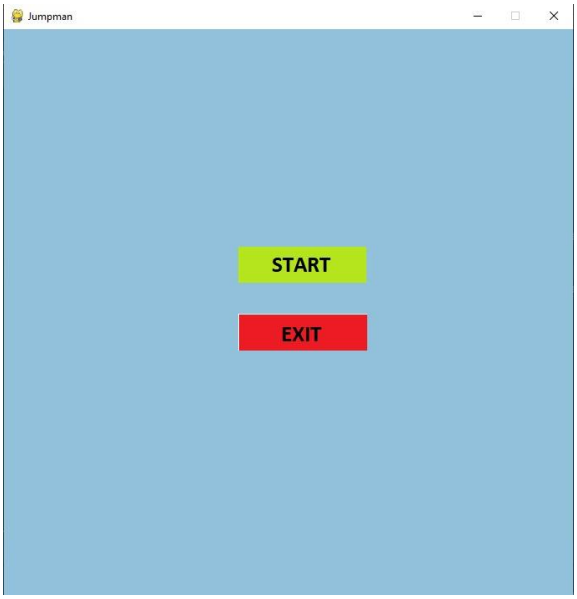
2.3 Features of Our Project

Our game is designed much simpler. The game has different aspects from the original. We tried to design a game that was as close to the original as possible but different.

3. USER MANUEL

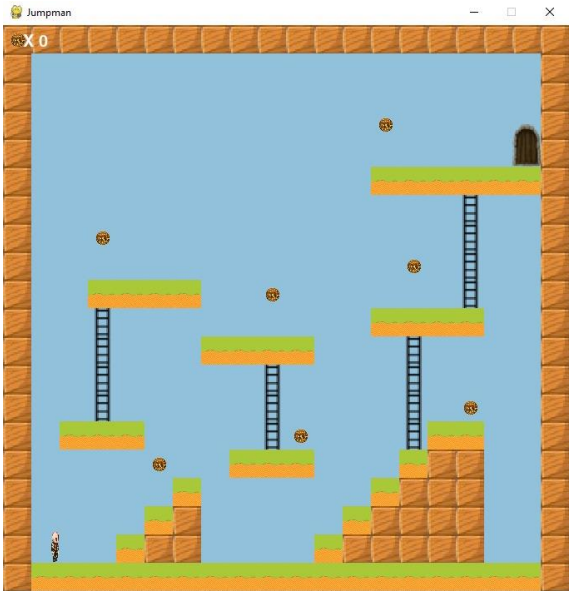
3.1 Menu Display Screen

This is view of menu display screen. There are 3 buttons here. Start, Exit. If you click the start button, the game starts. If you press the exit button, the game will not start and the screen will turn off. If you press the Handbook button, a screen opens with everything you need to know about the gameplay.



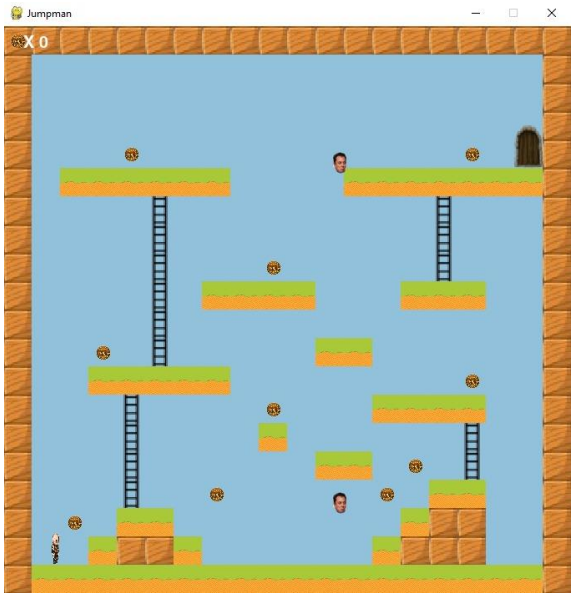
3.2 Easy Level Display Screen

In the easy level, the player can move around the platform, climb stairs and collect coins. If player can collect all the coins, do not fall from the ladder or gap and reach the exit door, player will move on to the next level.



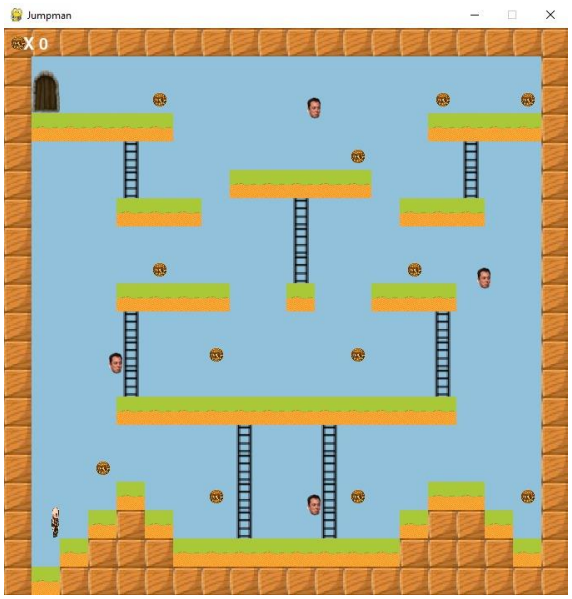
3.3 Normal Level Display Screen

In the normal level, the player can move around the platform, climb stairs and collect coins. In this level there is a enemy moving along the x-axis. If the player touches this enemy, player dies and the game is over. If player can collect all the coins, do not fall from the ladder or gap, reach the exit door and does not touch the enemy, player will move on to the next level.



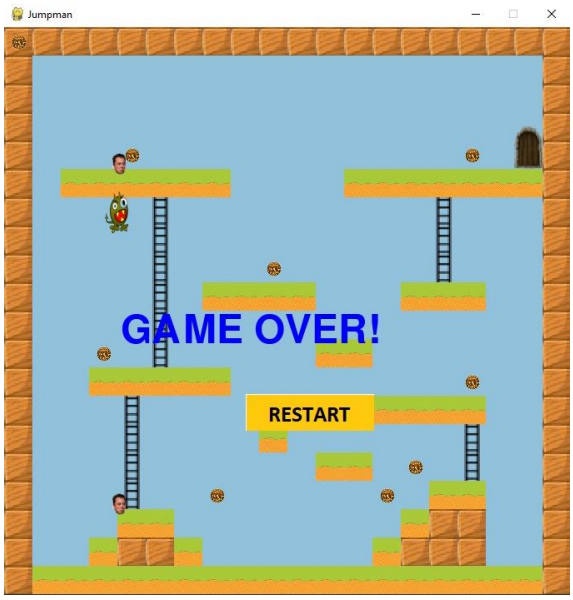
3.4 Hard Level Display Screen

In the hard level, the player can move around the platform, climb stairs and collect coins. In this level there are two enemies moving along the x-axis and y-axis. If the player touches these enemies, player dies and the game is over. If player can collect all the coins, do not fall from the ladder or gap, reach the exit door and does not touch the enemies, player will win the game.



3.5 Game Over Display Screen

In all levels, if the player falls from the ladder or the gap, the player dies and the game is over. In normal and hard levels, if the player falls from the ladder or the gap and touches the enemy, they die and the game is over.



4. CODE REVIEW

4.1 Main: (game.py)

- This class defines the logic of the game and how player input is taken etc.
- We run one instance of this class at the start of the game, and this instance manages the game for us.

4.1 Button Class:

- We initialized Button Class.
- This class defines all our buttons.
- A button has an image and a position associated with it.

4.2 World Class:

- We initialized World Class.
- This class defines all inanimate objects that we need to display on our world.
- Any object that is on the world and not a person, comes under this class (ex. Coins, Ladders etc.)
- Sets up the image and its position for all its child classes.

4.3 Player Class:

- We initialized Player Class.
- This class defines our player.
- We specialize the person by adding capabilities such as jump etc.

4.4 Enemy Class:

- We initialized Enemy Class.
- This class defines all our enemies.
- Our enemies will move at the determined levels along the x and y axes.

4.4.1 Enemy Class:

```
class Enemy(pygame.sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.image.load('img/enemy.png')
        self.rect = self.image.get_rect()
        self.rect.x = x
        self.rect.y = y
        self.move_direction = 1
        self.move_counter = 0

    def update(self):
        self.rect.x += self.move_direction
        self.move_counter += 1
        if abs(self.move_counter) > 50:
            self.move_direction *= -1
            self.move_counter *= -1
```

4.5 Platform Class:

- We initialized Platform Class.
- Used for the positions of the platforms from which the player climbs.

4.5.1 Platform Class:

```
class Platform(pygame. sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        img = pygame.image.load('img/platform.png')
        self.image = pygame.transform.scale(img, (tile_size, tile_size //
2))
        self.rect = self.image.get_rect()
        self.rect.x = x
        self.rect.y = y
```

4.6 Ladder Class:

- We initialized Ladder class.
- This class defines all our ladders in the game.
- We can add features such as ladder climb sounds etc. here.

4.7 Coin Class:

- We initialized Coin class.
- This class defines all our coins.
- Each coin will increase our score by an amount of 'value'.
- We animate each coin with 1 image.

4.7.1 Coin Class:

```
class Coin(pygame.sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        img = pygame.image.load('img/coin.png')
        self.image = pygame.transform.scale(img, (tile_size // 2,
tile_size // 2))
        self.rect = self.image.get_rect()
        self.rect.center = (x, y)
```

4.8 Exit Class:

- We initialized Exit Class.
- It was created so that the player can pass to the next level and win the game in the last level.

4.8.1 Exit Class:

```
class Exit(pygame.sprite.Sprite):
    def __init__(self, x, y):
        pygame.sprite.Sprite.__init__(self)
        img = pygame.image.load('img/exit.png')
        self.image = pygame.transform.scale(img, (tile_size, int(tile_size
* 1.5)))
        self.rect = self.image.get_rect()
        self.rect.x = x
        self.rect.y = y
```

5. WORK SHARING

Each member of the team will work together on all parts of the project.

- **Kani SUNGUR:** He is the team member who is going to organize the meetings. He determines the meeting time and which platform the meeting is going to be on. When the meeting time approaches, he connects with team and brings them together. He manages the order of speaking during the meeting and prevents tensions that may ocur. He is the team member who is taking the meeting notes that what was planned in the meeting and what we decided to do until next meeting. And he notes who did or did not do the tasks determined according to the previous meeting notes. Also he reports the key points discussed in the meeting.

- **Refiye Şehnaz YILDIRIM:** She is the person who is going to direct the team in the code part of the Project. She is the person who checks the fidelity of the project. She oversees the progress of the Project and assembles the Project. She is the person who prepares the reports of the project. She creates the reports by taking the meeting notes, learning the final version of the code, section and checking the meeting attendance and activity status.

REFERENCES:

1. https://en.wikipedia.org/wiki/Jumpman_(video_game)
2. https://mixkit.co/free-sound-effects/win/
3. https://www.playdosgames.com/online/classic-jumpman/#cheats
4. https://github.com/timmmmmb/jump
5. https://opensource.com/article/18/7/put-platforms-python-game
6. https://arcade.academy/examples/platform_tutorial/step_11.html
7. https://arcade.academy/examples/10_ladders_and_more.html
8. https://programmersought.com/article/77325259692/
9. https://inventwithpython.com/inventwithpython_3rd.pdf
10. https://inventwithpython.com/makinggames.pdf
11. https://github.com/erilyth/DonkeyKong-Pygame
12. https://github.com/codewmax/Pygame_Tutorials