COMPUTER GAMES REPORT

CSE420 - COMPUTER GAMES

DUE DATE (13.06.21)

1980'S JUMPMAN GAME

JUMPSTUDENT

171805030 Gülfide ÇALIŞKAN

181805028 Betül BAŞAN

History of Jumpman

Jumpman is a platform game written by Randy Glover and published by Epyx in 1983. It was first developed for the Atari 8-bit family, and versions were also released for the Commodore 64, Apple II, and IBM PC. The game received very favorable reviews when it was released and was a major hit for its publisher, Automated Simulations. It was so successful that the company renamed itself Epyx, formerly their brand for action titles like *Jumpman*. Re-creations on other platforms, and new levels for the original versions, continue to appear. *Jumpman* was published on diskette, but a version of the game with 12 new levels instead of 30 was released on cartridge as *Jumpman Junior*. It was available on the Atari 8-bit computers, Commodore 64, and ColecoVision.

Alien creatures have broken into the Jupiter headquarters and sabotaged all systems and hid bombs in different corners and angles, which can destroy the headquarters and the two adjoining buildings any moment. It all depends on you. As secret agent Jupiter Jumpman you have the speed and experience to cross the aliens' plans and thus defend the headquarters. In this comprehensive platform game, the bombs have to be collected in 30 levels. At the same time some creatures as bats, birds or aliens, but also robots and missiles disturb you. Also, collecting the bombs is a bit tricky. Either traps are triggered, i.e. ropes, ladders or platform parts disappear or fall down, or in some levels they only appear by this. The name of the level kind of contains directly what will happen! Seen as that, everything that moves should be avoided. Usually this platform game is played by jumping, but in some few levels alien creatures need to be shot. Jumpman was rather successful in 1983, so that there was a follower game Jumpman Junior in 1984.

Design

The levels are designed rather simply, i.e. most participants are unicoloured but the level altogether is multicoloured. Creatures, especially Jumpman, are animated. Simple souund effects as the jumping and running of Jumpman, movement of creatures, the missiles or bombs accompany the game. A short tune is played when you lose a life, at the end of a level and at the end of the game.

Development

Randy Glover was living in Foster City, California and had been experimenting with electronics when he saw his first computer in 1977 when he played *Star Trek* at a Berkeley university open house. This prompted him to purchase a Commodore PET in 1978, and then upgraded to a TRS-80 due to its support of a hard drive.

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. He visited a local computer store who had the TI-99/4A and Atari 400. He initially purchased the TI-99 due to its better keyboard, but when he learned the graphics were based on character set manipulation, he returned it the next day and purchased the Atari.

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.

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. The Atari version used a classic bad-sector method of preventing copying, but this had little effect on piracy.

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. Two of its levels (Dumbwaiter and Electroshock Traps) were turned into Sreddal ("Ladders" backwards) and Fire! Fire! on the latter. The C64 version was later ported to the ColecoVision, which used the C64 levels.

Glover continued working at Epyx, working on the little-known *Lunar Outpost* and the swimming section of *Summer Games*. He remained at the company for about two years before returning to the cash register business.

Technical details
Movement was controlled through the collision detection system of the Atari's player/missile graphics hardware. This system looks for overlap between the sprites and the background, setting registers that indicate which sprite had touched which color.

Glover separated the Jumpman sprite into two parts, the body and the feet. By examining which of these collided, the engine could determine which direction to move. For instance, if both the body and feet collided with the same color, it must be a wall, and the Jumpman should stop moving. If there was no collision with either his feet or body, Jumpman is unsupported and should fall down the screen. Variations on these allowed support for ramps, ropes, and other features.

This not only saved processing time comparing the player location to an in-memory description of the map, but also meant that maps could be created simply by drawing with them and experimenting with the results in the game.

Hints

There are 5 game variants:

- 1. Beginner (level 1-8)
- 2. Advance (level 9-18)
- 3. Professional (level 19-30)
- 4. All levels
- 5. Random level (1st random level is always "Robots 2")

After completing the game variants 1-3 the Jupiter headquarter appears.

Jumpman's speed is preset for each level and can be seen next to SPEED. But it can be changed by pressing the keys <1> to <8> before the Jumpan appears in the level: 1 is the fastest and 8 the slowest speed.

You start with 7 lives, every 10.000 points you get an extra life.

Each collected bomb gets you 100 points, except for the level "Grand Puzzle" (here: 500 points per bomb). Shooting creatures also gets you points (not in all levels possible!).

In the game there are 2 types of ropes which Jumpman automatically climbs up or down. You can also jump from or onto the ropes!

- Green ropes: He only climbs upwards.
- Blue ropes: He only climbs downwards.

Caution: Jumpman's speed also effects the speed of the creatures.

Controls

↑ / ↓ : climb up/down ladders

← / → : walk left/right

⬅ / ➡ : jump left/right

⬆ : jump up

Also the following keys have special meaning:

Esc: Pause game

Ctrl + S: Disable or enable sound

Ctrl + R: Aborts current level

Alt + X: Exits the game

Solution

The solution for some levels can be viewed in the video section or in the video at the C64 game video archive.

Cheats

The version by "Remember" has different cheat modes (unlimited lives, sprite collision off, jump level).

Critics From C64-Wiki

Jodigi: "A nice game. One might think that entering and leaving the ladders is a bit problematic. But maybe this is intended so."

Worf: "A really bad game! Only the great number of levels can be mentioned positively. A real B-game."

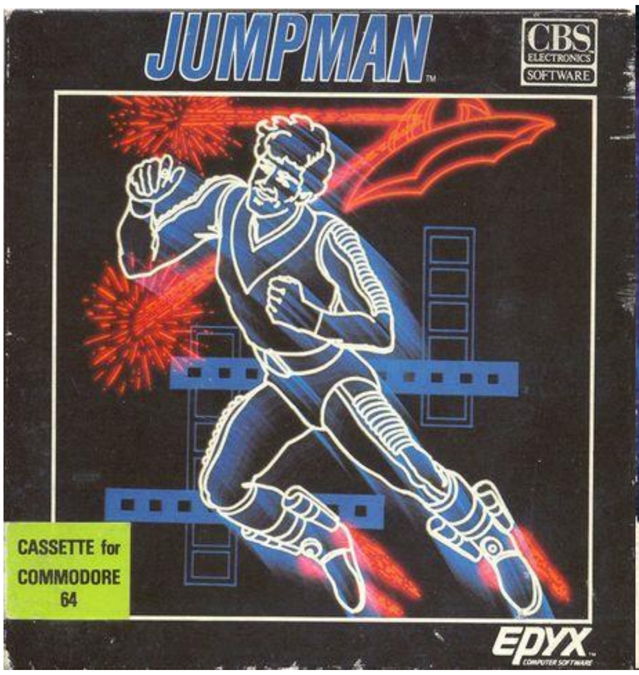
TheRyk: "Okay, an early game. But compared to Lode Runner which is from 1983 as well, Jumpman is really poor. Being stuck in ladders again and again is really annoying. Apart from mere nostalgic reasons, nobody needs that. 3 out of 10 points."

Robotron2084: "Jumpman does not look exhilarating, but plays really great. For me the game has the same cult status as e.g. Space Taxi, Boulder Dash or H.E.R.O.. Not that I would give those games 10 points, but for me these are all essential cornerstones of the (C64) computer game history. Everything else is blasphemy. 7 out of 10 points."

Camailleon: "It would be a nice platformer if it wasn't for the unfair situations as e.g. that the enemies are not set back after losing a life, so that it often happens that after reappearing you lose another screen life. Another mistake is that enemies can appear from the side where you currently have to walk along the border. Then you have no chance to avoid the enemy."

NSH: "I love this game, one of my favorites. To me the problems are just part of what makes it challenging- avoid getting stuck on ladders, avoid edges of the screen when not all enemy bullets are visible, and have a chuckle when you die at an inopportune moment such that you reappear on an enemy. Surviving the Grand Loop is a great and satisfying achievement. *9 out of 10* points."

Cover **Advertisement**



Reception

Jumpman became a best-seller for Epyx, selling about 40,000 copies on the Atari and C64 until 1987, reaching somewhere between #3 and #6 on the then-current *Billboard* top 100 games chart. Sales were hindered by the release of *Miner 2049er* only a few months earlier, which held the #1 spot at that time.

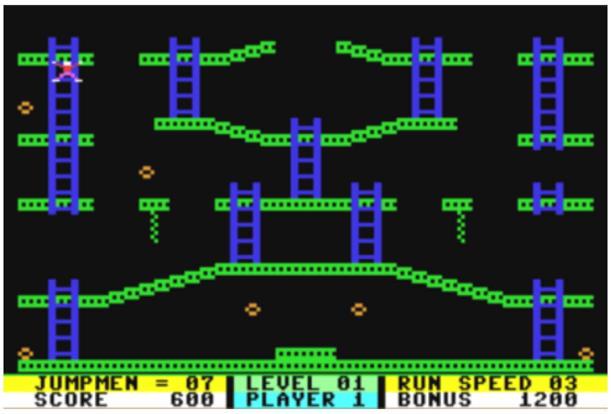
Softline in 1983 liked *Jumpman*, calling it "wonderfully addicting" and stating that it was as high-quality as Epyx's *Dunjonquest* games. The magazine cited its large number of levels ("Not one screen faster and harder each time; not ten screens three times; but thirty screens, one at a time"), and concluded that "it's bound to be a hit". *Compute!* awarded it a lengthy review. They point out that it might be dismissed as yet another platform game, but goes on to state that "Jumpman easily conquers that skepticism and establishes itself as a software classic." They also note the variety of clever level designs that makes each map unique. They go on to compare it to *Miner 2049er* and suggested *Jumpman* is "much, much more."

In 1984 *Softline* readers named the game the seventh most-popular Atari program of 1983, and it received a Certificate of Merit in the category of "1984 Best Computer Action Game" at the 5th annual Arkie Awards.

K-Power rated the Commodore 64 version of *Jumpman* 7 points out of 10. The magazine stated that the game "has very good—not great—graphics, color, and sound. But because it's so enjoyable to play, it will be a long time before it's put away." Stating that "the care that goes into its products is obvious in Jumpman", *The Commodore 64 Home Companion* wrote that "it's really 30 games in one, with seemingly endless variants on the simple jumping theme to keep you interested".

Beginner: Easy Does It

Level 1: Easy Does It



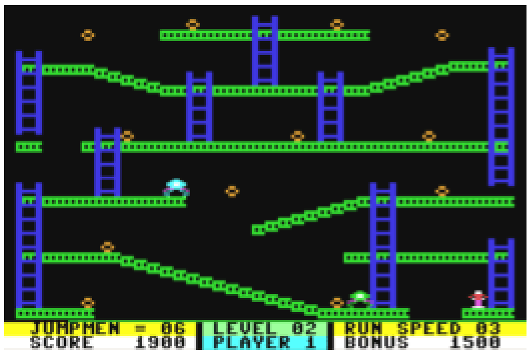
Here, we've only got one Alienator bullet to deal with. Not too bad. The level is pretty straightforward, but, quickly it starts setting the tone for the game. First off, hitting a bomb may have unexpected results. In this case, hitting a specific bomb will remove a piece of a ladder. It's not enough to trap you from your destination, but, it does change the flow of gameplay.

You'll also notice that there's jumps of varying widths – Jumpman can make the span of every jump on this level, but that's not going to be the case for later levels.

One of the level elements is a "up rope". With ladders, you can go up or down with them, and you control your movement in the process. With ropes, they come in two flavors – up, or down. If you jump on the up rope, Jumpman begins climbing up them. The only real control you have is that you can jump off them – you can't speed up or slow down.

And, you'll discover your first Jumpman frustration: the trip and fall. In the center of the bottom most girder is a platform. If you step off the platform, Jumpman… well, he's not graceful. He trips, falls flat on his face, and you die.

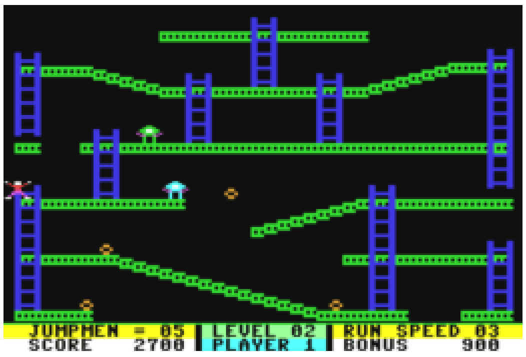
Beginner: Robots



Level 2: Robots (at the beginning)On this level we won't find any Alienator bullets. Good thing, too – we've got a new element to deal with.

The robots are a re-occuring theme in the game – a few levels (like the Grand Puzzle) levels occur in each of the difficulty sets of Jumpman, each with a new variation. Robots is one of the more interesting (to me) variations, because they slowly become more intelligent. Additionally, the make really cool noises in Robots II and III ☺

The robots are static, until you touch a bomb. Once you do that, they go into action. They run on a pre-determined route, and each bomb triggers them to move to the next section of their route. So, part of the trick for the player is finding out what the route it.

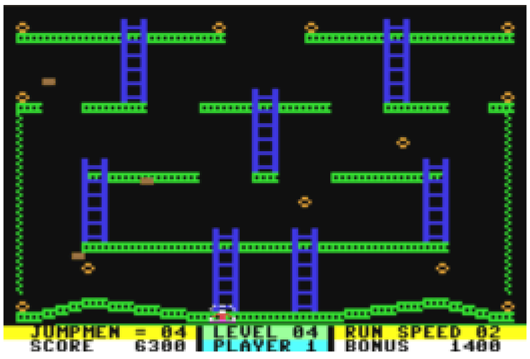


Level 2: Robots (almost done – notice the change in robot location)The robots are just tall and wide enough that you can't jump over one while it's in standing still – but, if it's in motion, headed your way, you can clear them if you're careful.

Another new element begins to show up at this point, too: efficiency of motion. You'll notice a Bonus counter in the lower right corner. The faster you complete a level, the larger your bonus in Jumpman. In Jumpman, Jr, it becomes a bonus for every life you have left – which removes a little bit of incentive for running at faster speeds.

The efficiency comes in play once you discover Jumpman's ability to catch a girder in mid-jump and climb up. The two bombs at the top are more easily reached if you just take the flying jump from the angled girder and catch the top-most girder, climb up, grab the bomb, run across, and jump off and land – you'll land just inside Jumpman's fall tolerances.

Beginner: Jumping Blocks

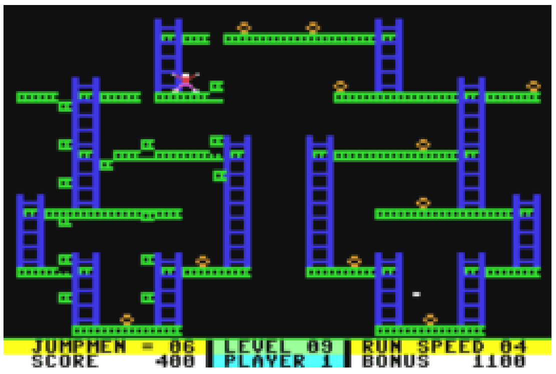


Level 3: Jumping BlocksOK, this one is weird. You've got a glowing, oversized version of the Alienator bullets. You're first instinct is to avoid them like the plague – they act exactly like the bullets, accelerating at you once they line-up with you.

But, the first time you make a mistake (because there's always a good number of them on screen), you discover their ability isn't going to directly kill you. Instead, they cause your jump boots to malfunction, and you get sent jumping in a random direction. Not bad, at first, until you're trying to pick off the bombs at the edge of the screen, or climbing the rope.

Intermediate: Look Out Below

Intermediate Level 1: Look Out Below, about midway through the level.This time, the Alienator bullets have returned to keep things moving along. See, if you have an infinite amount of time to very carefully step and be ready to react, this wouldn't be too hard. Throwing the bullet in there means sometimes you've got to just move and react to what happens – and this one has some surprises!

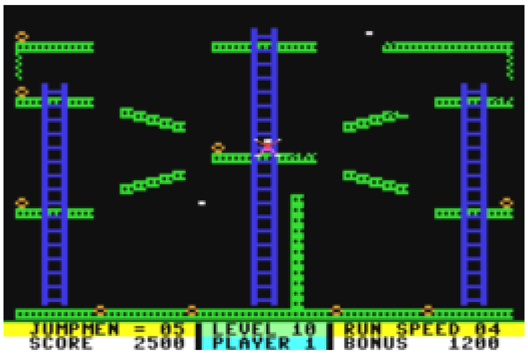


Now, every time you touch a bomb a part of the level falls. It's pre-scripted which parts fall, so once you know the pattern, it's a bit easier.

But, two things happen. First, the entire level starts turning into a trip hazard. You can see how there's blocks sticking up – they fell from the level above, and get stuck on the level below. Second, it's not consistent how far up they stick, or, sometimes as they fall they accidentally leave behind dirty clone versions of themselves.

Intermediate: Hot Foot

Intermediate level 2: Hot Foot, part of the way through the level.This time the new game mechanic is a reaction to your motions. Every time you jump, the floor space where you boots were explode! It doesn't do any damage to the player, fortunately. But, it does leave holes in the girders for you to trip on if you aren't careful. And, we've got an Alienator bullet (or two) running around to keep us jumping.



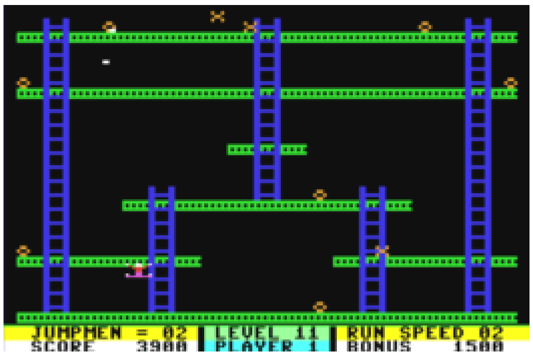
This level holds an important thing to know: order of operations in Jumpman can render a level unfinishable. In this case, the four bombs at the bottom of the level are the problem. Touching the one second to the left will encase the one that's all the way to the left in a piece of girder that can't be removed. At first it might seem that you can blow them apart by jumping up and down on them. And you can… sometimes. Some other times, well, it's just stuck.

There's a number of levels that have small design flaws like this – fortunately, it's not frequent.

Intermediate: Runaway

Yep, time for something new. Moving bombs. Oh, bloody heck.

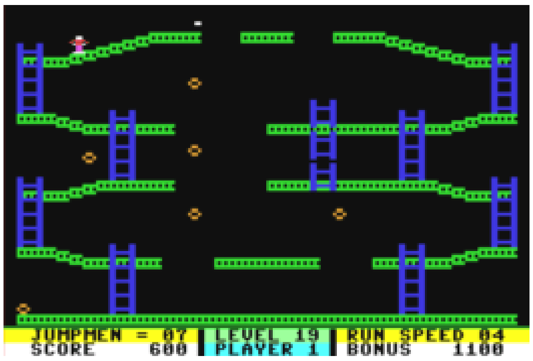
Intermediate Level 3: RunawayOn this one, the bombs semi-randomly decide to pack up and move. In the picture, you can see the things that look like X's, but are the color of the bombs? Those are bombs that are on the move. Fortunately, you can catch them in mid air while they're on the move. Unfortunately, they also move at the same speed as you, so, you have to figure out their paths and catch them head on.



And, of course, we've got a couple of Alienator bullets trying to chase Jumpman down. And, as you can see in the screenshot, sometimes they succeed – Jumpman is falling to his death in this screenshot.

Advanced: Ladder Challenge

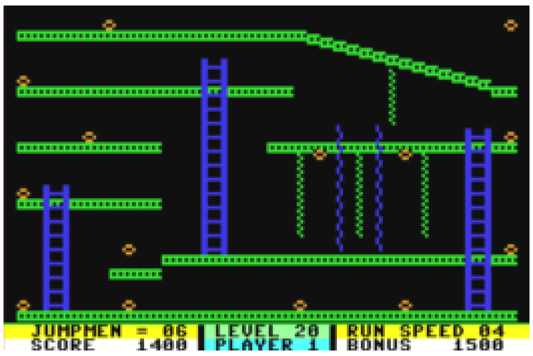
Advance Level 1: Ladder Challenge, about half way doneThis is a level where you can tell the game pushes the hardware (at the time) just a little hard in some places. In the center is a moving ladder. Which, well, doesn't look too impressive when you start the level – it moves around on a pre-set path, and if you run into it, nothing much happens.



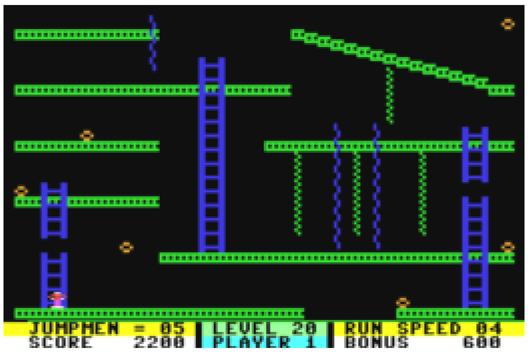
It's collecting the bombs that reveal it's true purpose. As you collect them, you'll see parts of the level disappear, and you'll see new bombs appear, floating in a spot that Jumpman can't quite jump high enough to get. Instead, you've got to learn to ride the ladder, working your way up and down as it goes along it's route (and, of course, avoid an Alienator bullet.)

Advanced: Figurit

Advanced Level 2: Figurit, at the beginningFigurit would hold the vote for "level most likely to make you throw your controller across the room, and the go get another one because you MUST beat it." This level takes the ability to re-configure the levels based on grabbing a bomb to new extremes.



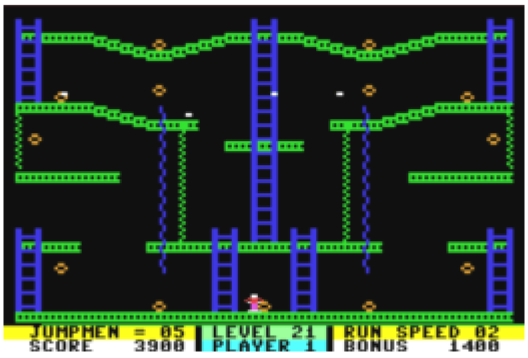
Also, while they appear in other levels before this, this is the first screenshot with a "down rope". Down ropes have a problem – just like Jumpman can trip and fall from a small ledge, going off the bottom of a down rope often ends up dropping Jumpman a little too far (I'll get into WHY that happens in the next article on controls and Jumpman's reactions – Randy was the one that explained the secret to me.) The best course if a down rope doesn't terminate exactly onto a girder is to jump off.



Advanced Level 2: Figurit, and there's no way left to figurit out.This is also a level that has an order of operations problem – pick up the bombs in the wrong order, and you're screwed. It's also got a level design issue: it's almost too random what's going to happen when you grab bombs. It wouldn't be so bad if there weren't so many. BUT – the name "Figurit" sort of hints you're not going to have an easy time on this one.

In the screenshot above, you can see the "stuck" configuration for the level – grabbing the wrong two bombs is all it takes to render the level unplayable (though, that's also partially because of bad luck in this case – I died while messing around trying to get screenshots at the same time). Grabbing one of the bombs ends up removing a small chunk of ladder – you can't jump quite high enough to grab the next piece. Grabbing the other bomb ends up removing the ledge you could have jumped onto instead of climbing the ladder. There's no way to finish the level.

Advanced: Jump-n-Run



Advanced Level 3: Jump-N-RunWhile complained about the previous level, this level actually makes up for it. There's a number of Alienator bullets after you (I see four on the screen in this picture, but, I believe it maxes out at five) that keep things moving. The level is reactive, reconfiguring small bits here and there as you grab bombs, though not ever bomb causes the level to change.

And it's just got amazing flow. The up / down rope combo, for instance, acts as a nice express way for Jumpman – climbing up or down the ropes moves faster than climbing a ladder, so they become good tools for dealing with the large number of bullets.

For me, it's the best of the you-vs-Alienator bullet levels. There's enough of them to make it interesting, but Randy Glover had given you just enough tools to keep away from them – as long as you never stopped moving.

An Interview with Randy Glover about Jumpman

I read somewhere that the Donkey Kong arcade was the beginning of it all? Please tell me a little history of Jumpman.

"Yes. Donkey Kong was out in the arcades and although it was not my favorite arcade game at the time it is what caused me to choose the format I did for Jumpman."

There's a PC-Booter version of Jumpman in CGA graphics out there. It looks like IBM had something to do with it. Do you know anything about that? Is that the official PC-Version?

"I have seen that version. While I worked at Epyx there were no discussions of a PC version. I assume that someone must have struck out on their own for that one."

Is it luck or cleverness, that you now own the 'Jumpman' name?

"Luck I guess. Contract clause called for the name rights to return to me 7 years after production/sales of the product ceased. Was that a standard clause or not? That I cannot answer so I have to call it luck. I would have agreed to about anything back then. :-)"

Was Jumpman already 100% finished when you contacted Epyx? I mean did they influence the game somehow?

"The program was about 85% when Epyx purchased it. At that time the people at Epyx had little experience in action games so they just let me finish on my own terms. They had little to no imput on any part of the game design."

Did you really do it all on your own? "Yes."

Where did you get all these awesome ideas from? Each of the 30 levels seems totally unique...

"Where do ideas come from? Always a good question. I cannot remember much in the area of specific inspiration for most things in the game. A lot of the ideas were based on working the "game engine"."

Did you have even more ideas which didn't make it in the game? Stuff you dropped, because it was not realizable or not playable?

"Many. Most dropped because just to difficult to implement. The most memorable one being a shark tank in the middle in which Jumpman must swim (and avoid sharks)."

I think I read somewhere, that each level had it's own source code, is that true?

"80% of the level was run by the game generator. In order to ad each levels special flair unique code was created for each."

When playing your own game, did you manage to finish all 12 levels in "Advanced" mode? I think some of them redefine the word "Advanced" a bit... :-)

"Yes ... but of course, I knew all the angles and lots of practice."

Did you know that you created such a hit, when you were done with it?

"Heck no. I was in it for the fun of writing the code and creating a game. It wasn't till much later that I realized the potential for a hit game."

What do you think about all these people still playing Jumpman today?

"I have no idea how many might still be playing it. I do know, that from previous contacts, many have a fondness for it as they played it when they were younger."

JUMPSTUDENT(JUMPMAN)

We created an enhanced version of 1980's Jumpman.

Program Description: The student is trying to go to school. There are two things that prevent the student from going to school: the phone and the lava. The student needs to stay away from these two things and collect the books on the road and reach the school. When all three levels are completed, the task is complete. Good luck!

CONTROLS:

DURING GAME PLAY:

LEFT ARROW	Student moves left
RIGHT ARROW	Student moves right
SPACE BAR	Student jumps

GAME OVER/YOU WIN MENU:

RESTART	Select restart and the game restarts

START MENU:

PLAY	Select play button and the game plays
EXIT	Select exit button and the screen closes

LEVEL EDITOR:

You can create your own level by clicking the tiles one after the other and selecting the object you want (such as phone, school) in level editor.

1 = dirt blocks

2 = grass blocks

3 = phone

4 = horizontally moving platform

5 = vertically moving platform

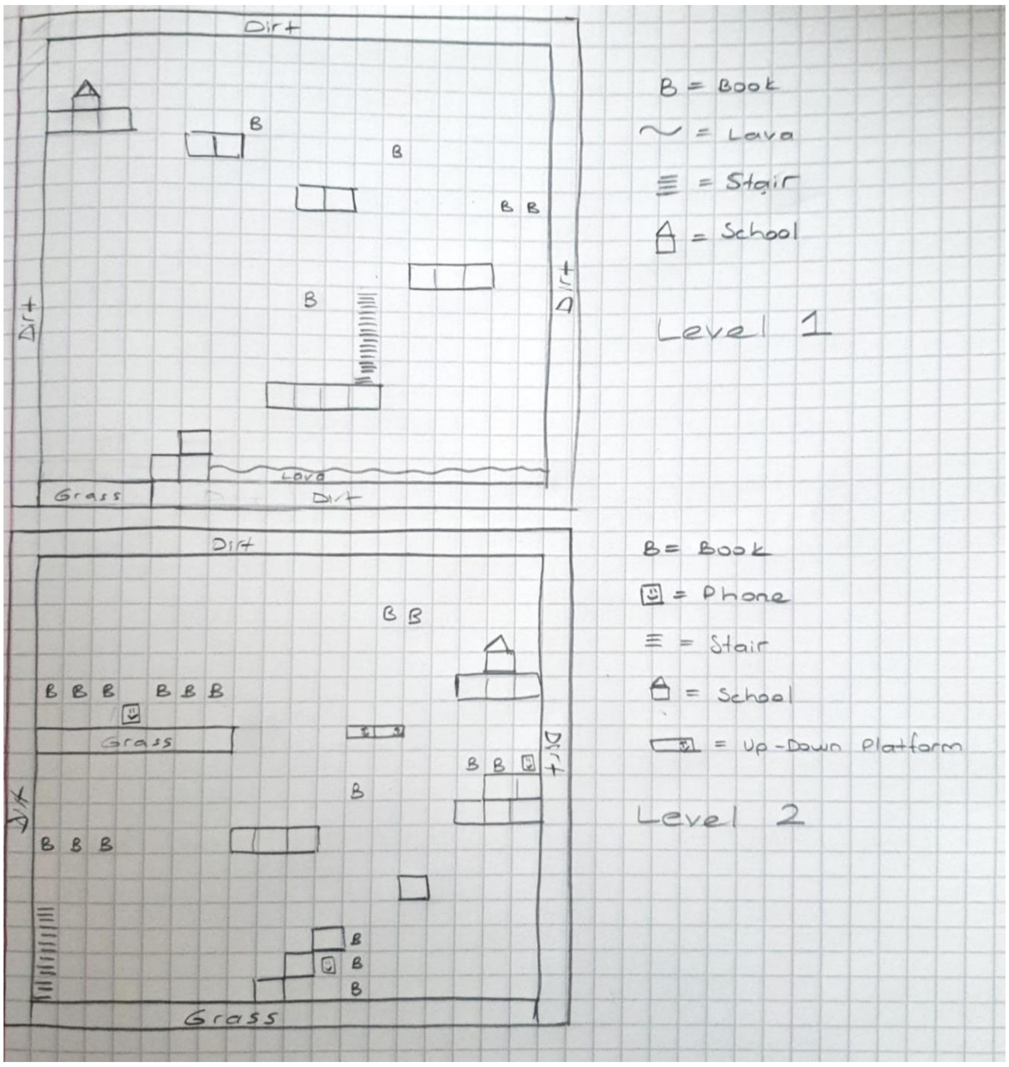
6 = lava

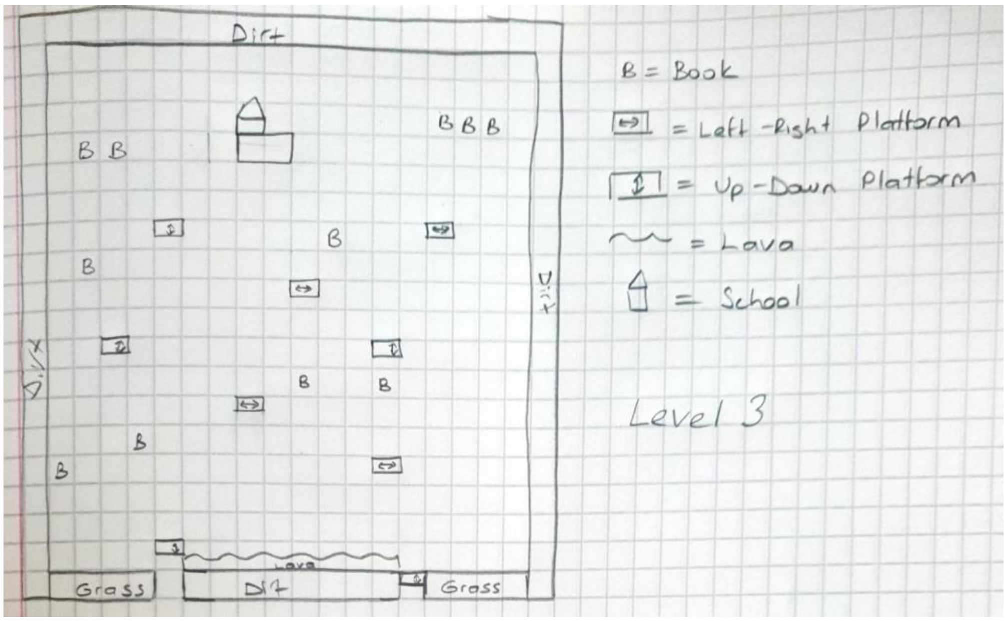
7 = book

8 = stair

9 = school

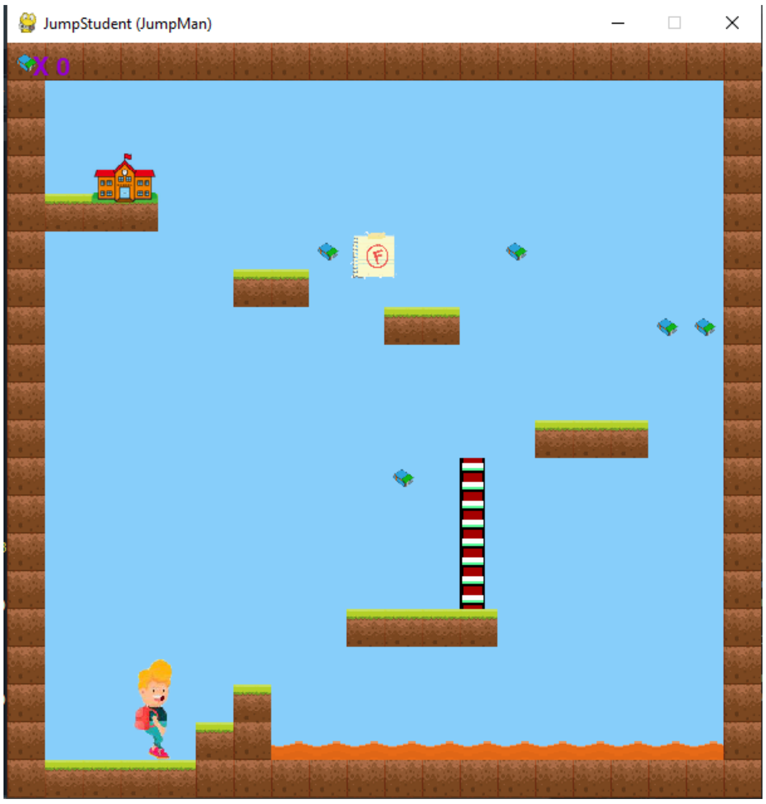
Sketches:


Dirt
B = Book
~ = Lava
≡ = Stair
= School
Level 1
Grass
Lava
Dirt
Dirt
B = Book
= Phone
≡ = Stair
= School
= Up-Down Platform
Level 2
Grass
Grass

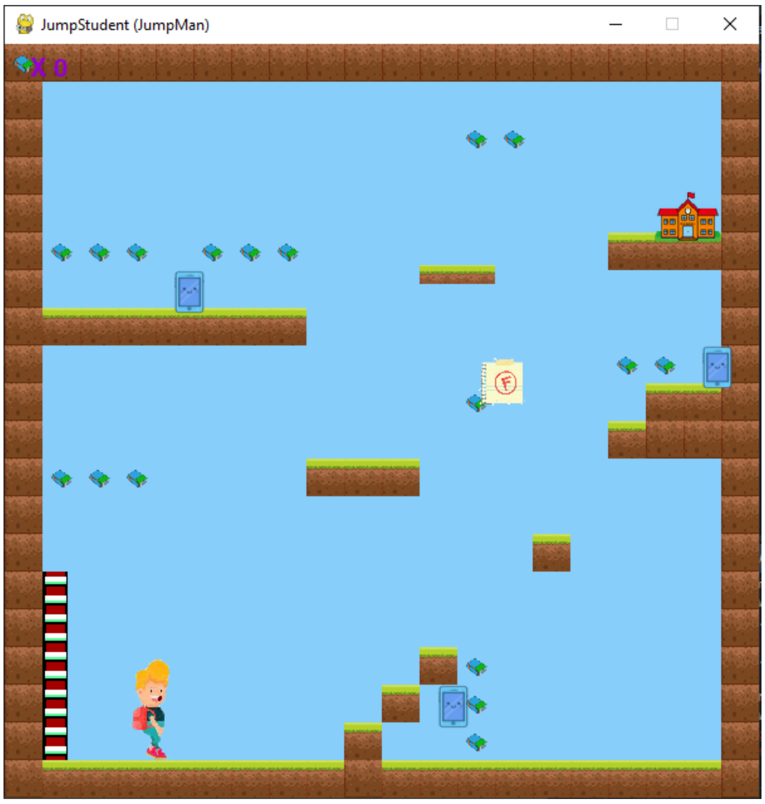

Dirt
B = Book
= Left-Right Platform
= Up-Down Platform
~ = Lava
= School
Level 3
Grass
Lava
Dirt
Grass

Levels:

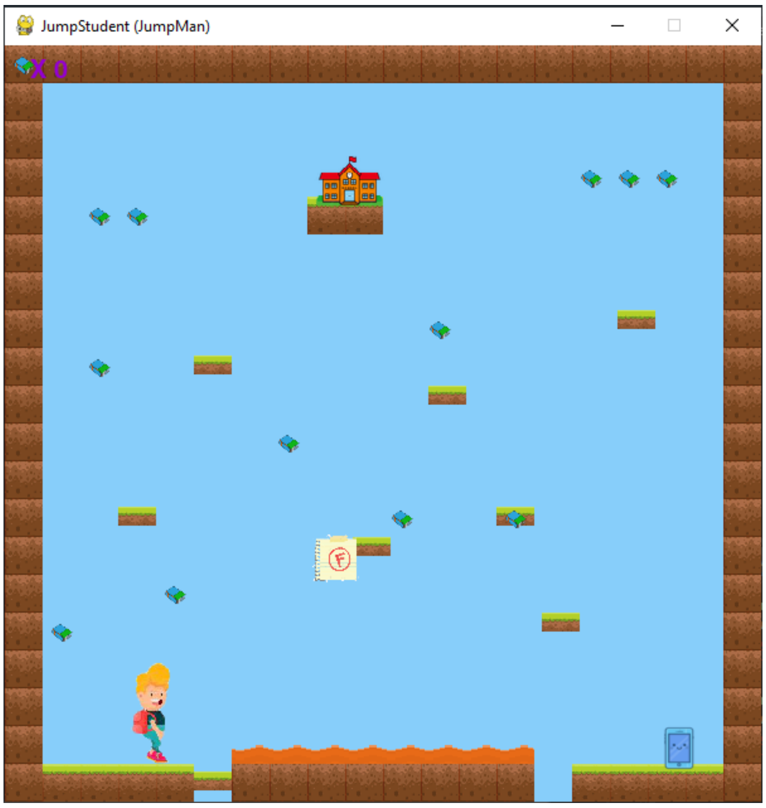
Level 1:



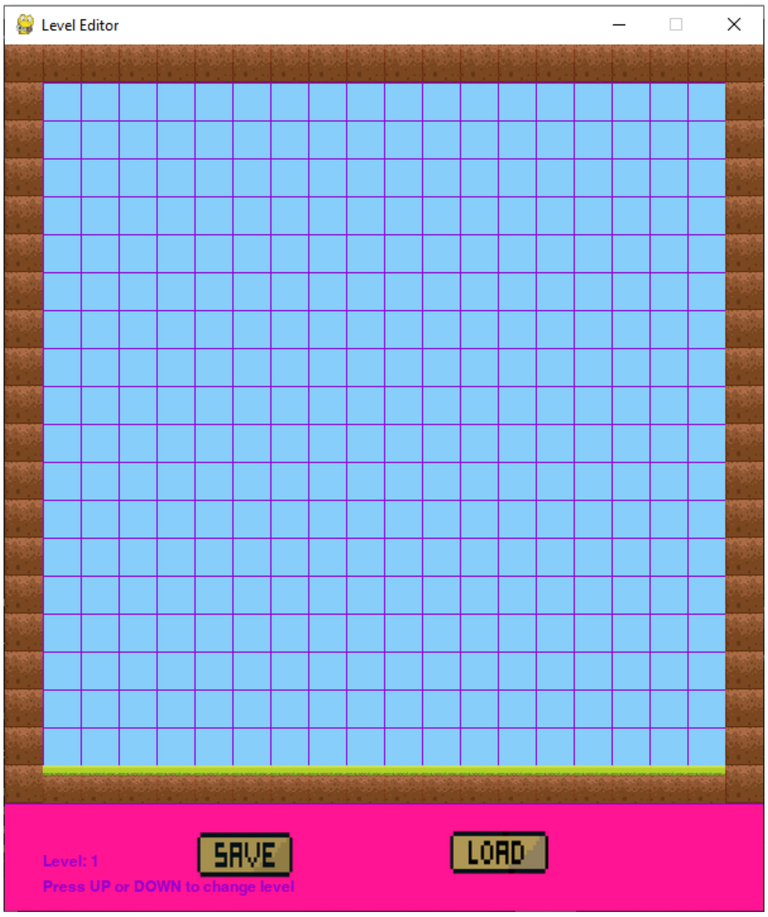
Level 2:



Level 3:



Level Editor:



Level1_data, level2_data and level3_data are Python lists.

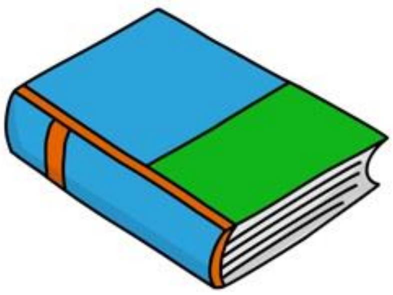
We used level1_data, level2_data and level3_data for creating Python lists that we created levels.

Our Objects We Used:

Book:

The book was created to be collected like a piece of gold.

The number of books collected determines the score.



Phone:

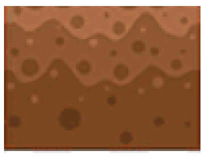
The phone has been used as an enemy. When you touch the phone, you die.

Ghost:

Ghost appears on the screen as you die when you hit the F note and the phone.

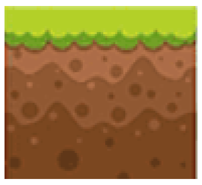
Dirt:

The dirt was used as a platform.



Grass:

The grass was used as a platform.



School:

The school was used as the place to be reached in the game.

When you arrive at the school, you move on to the next level.

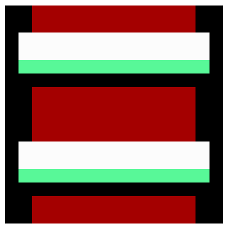
Student:

The student was used as an player. The student's goal is to

arrive at school without touching the phone and the F grade.

Stair:

The student reaches the platforms above by climbing the ladder.



NoteF:

The F grade was used as an enemy to be avoided for the student.

If the student touches the F grade, he dies.

CODE REPORT:

main.py:

FUNCTIONS	EXPLANATIONS
Pygame.font.SysFont()	Define font type and font size
Pygame.image.load()	Load image

pygame.mixer.music.load()
pygame.mixer.music.play()
pygame.mixer.Sound()
book_fx.set_volume()
	Load music,play music and set volume

def draw_text(text, font, text_col, x, y)	Define coordinate and font for writing text
reset_level(level)	Function to reset level, load in level data and create world

phone_group = pygame.sprite.Group()
platform_group = pygame.sprite.Group()
lava_group = pygame.sprite.Group()
stair_group = pygame.sprite.Group()
book_group = pygame.sprite.Group()
school_group = pygame.sprite.Group()
noteF_group = pygame.sprite.Group()
	We defined a container class to hold and manage multiple Sprite objects.

score_book = Book(tile_size // 2, tile_size // 2)
book_group.add(score_book)
	We created dummy book for showing the score

if path.exists(f'level{level}_data'):
    pickle_in = open(f'level{level}_data', 'rb')
    world_data = pickle.load(pickle_in)
world = World(world_data)
	We loaded in level data and created world

restart_button = Button(screen_width // 2 - 50, screen_height // 2 + 100, restart_img)	We created buttons like restart button.

run = True
while run:
	We checked main menu and game over. We updated the score. We defined if student has died and has completed level in this loop.

CLASSES	
Student Class	**Define our player, get keypresses, handle animation, add gravity, check for collision, update student coordinates, draw student onto screen**

key = pygame.key.get_pressed()
if key[pygame.K_LEFT]:
    dx -= 5
    self.counter += 1
    self.direction = -1
	Example of defining one of the keypresses

self.vel_y += 1
if self.vel_y > 10:
    self.vel_y = 10
dy += self.vel_y
	Add gravity

self.in_air = True for tile in world.tile_list:	In this loop we checked for collision in x and y direction and jumping, falling.
if pygame.sprite.spritecollide(self, phone_group, False):	We checked for collision with other objects like phone.

game_over = -1
game_over_fx.play()	

def reset(self, x, y):	We loaded dead image and guy images. We loaded the attributes our student can do(jumping,climb,direction...)
World Class	
def __init__(self, data): #Codes	We coded our tiles by numbering them according to the objects in our game.

if tile == 3:
phone = Phone(col_count * tile_size, row_count * tile_size)
phone_group.add(phone)
	Our one of the tile is equal to 3 and that means that we created phone object.

Platform Class	
def __init__(self, x, y, move_x, move_y)	We loaded image of platform. We loaded its movement and direction.
def update(self)	We set the movement we loaded.

book.py:

FUNCTIONS	EXPLANATIONS

class Book(pygame.sprite.Sprite):
def __init__(self, x, y):
	We loaded the book image. We scaled the book.

button.py:

FUNCTIONS	EXPLANATIONS

class Button():
def __init__(self, x, y, image):
#Codes
def draw(self):
#Codes
	We got mouse position, checked mouseover and clicked conditions, drew button.

lava.py:

FUNCTIONS	EXPLANATIONS

class Lava(pygame.sprite.Sprite):
def __init__(self, x, y):
	We loaded the lava image. We scaled the lava.

noteF.py:

FUNCTIONS	EXPLANATIONS

class NoteF(pygame.sprite.Sprite):
def __init__(self):
#Codes
def update(self):
#Codes
	We loaded the F note image. We scaled the note F image. We used random library to set the range of note.

phone.py:

FUNCTIONS	EXPLANATIONS

class Phone(pygame.sprite.Sprite):
    def __init__(self, x, y):
    #Codes
    def update(self):
     #Codes
	We loaded the phone image. We set the direction of movement.

school.py:

FUNCTIONS	EXPLANATIONS

class School(pygame.sprite.Sprite):
       def __init__(self, x, y):
        #Codes
	We loaded the school image. We scaled the school.

stair.py:

FUNCTIONS	EXPLANATIONS

class Stair(pygame.sprite.Sprite):
       def __init__(self, x, y):
	We loaded the stair image. We scaled the stair.

settings.py:

FUNCTIONS	EXPLANATIONS

clock = pygame.time.Clock()
fps = 60
	We created an object to help track time.
	We defined standard fps for games.

screen_width = 600
screen_height = 600
 	
screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption('JumpStudent (JumpMan)')
	We defined our screen width and height.
	We set the name of game window.

tile_size = 30
game_over = 0
main_menu = True
level = 1
max_levels = 3
score = 0
	We defined game variables.

darkviolet = (148,0,211)	We defined one of the colors we used.
bg_img = pygame.image.load('img/bg.jpg')	We loaded one of the images.

book_fx = pygame.mixer.Sound('musics/book.wav')
book_fx.set_volume(0.5)
	We defined the sound of book. We set its volume.

leveleditor.py:

FUNCTIONS	EXPLANATIONS

clock = pygame.time.Clock()
fps = 60 #Standard fps
	We created an object to help track time. We defined standard fps for games.

tile_Size = 30
cols = 20
margin = 100
screen_width = tile_Size * cols
screen_height = (tile_Size * cols) + margin
 	
screen =
pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption('Level Editor')
	We set game window. We set the name of window as Level Editor.

dirt_img = pygame.image.load('img/dirt.png')	We loaded images.This is one of the example of loading images.
deeppink = (255,20,147)	We defined one of the colors we used.
font = pygame.font.SysFont('ravie', 18)	We created a Font object from the system fonts

world_data = []
for row in range(20):
    r = [0] * 20
    world_data.append(r)
	We created empty tile list.

for tile in range(0, 20):
    world_data[19][tile] = 2
    world_data[0][tile] = 1
    world_data[tile][0] = 1
    world_data[tile][19] = 1
	We created boundary using cols.

def draw_text(text, font, text_col, x, y):
    #Codes
def draw_grid():
    #Codes
	We created a function to output text onto the screen.

def draw_world():
if world_data[row][col] == 6:
     #lava
     img = pygame.transform.scale(lava_img, (tile_Size, tile_Size // 2))
     screen.blit(img, (col * tile_Size, row * tile_Size + (tile_Size // 2)))
	We coded our tiles by numbering them according to the objects in our game. Our one of the tile is equal to 6 and that means that we created lava object.

class Button():
    def __init__(self, x, y, image):
     #Codes
    def draw(self):
	We got mouse position, checked mouseover and clicked conditions, drew button.

save_button = Button(screen_width // 2 - 150, screen_height - 80, save_img)	We created load and save buttons to save or load levels

run = True
while run:
	We drew background , loaded and saved levels, showed the grid and draw the level tiles. We use draw_text function to show the current level. We handled events(Quit game, Mouseclicks to change tiles...). We updated game display window.

Other References:

https://en.wikipedia.org/wiki/Jumpman_(video_game)

https://www.c64-wiki.com/wiki/Jumpman

https://www.old-games.com/download/2382/jumpman

http://www.icfs.de/english/jumpman.html

http://programarcadegames.com/index.php?&chapter=example_code_longer_examples

https://inventwithpython.com/makinggames.pdf

https://ptgmedia.pearsoncmg.com/images/9781509304523/downloads/9781509304523_16_Python.pdf

https://www.sgo.fi/~j/baylie/McGugan%20-%20Beginning%20Game%20Development%20with%20Python%20and%20Pygame%20(Apress,%202007).pdf

https://www.youtube.com/channel/UCNaPQ5uLX5iIEHUCLmfAgKg

https://www.midnightryder.com/deconstructing-a-classic-jumpman-part-2/