

A Minor Project Report on
Flappy Bird Using Python

A dissertation submitted for 7th semester minor project



Submitted By

1901206086

Sourav Patnaik

1901206040

Abhijeet Kumar Sahoo

1901206089

Subhrajit Bastia

1901206042

Abinash Nayak

Under The Supervision Of

Er. Abhipsa Pattanaik

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AJAY BINAY INSTITUTE OF TECHNOLOGY
CUTTACK-753014
(2019– 2023) BATCH



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AJAY BINAY INSTITUTE OF TECHNOLOGY, CUTTACK

DECLARATION

We do hereby declare that the project report entitled, “**Flappy Bird Using Python**” submitted in the Department of Computer science and Engineering, Ajay Binay Institute of Technology, Cuttack of Biju Patnaik University of Technology, Odisha in partial fulfilment of requirement for the award of 7th semester BTech Degree in Computer science and Engineering is an authentic work carried out by us during 2022-2023 under the supervision of Er. ABHIPSA PATTANAIK. The matter presented in this report has not been submitted by us in any other University/Institute for the award of BTech Degree.

Submitted By

1901206086

Sourav Patnaik

1901206040

Abhijeet Kumar Sahoo

1901206089

Subhrajit Bastia

1901206042

Abinash Nayak



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AJAY BINAY INSTITUTE OF TECHNOLOGY, CUTTACK

CERTIFICATE OF APPROVAL

The project report named “Flappy Bird Using Python” is a bonafied work carried out by

1901206086

Sourav Patnaik

1901206040

Abhijeet Kumar Sahoo

1901206089

Subhrajit Bastia

1901206042

Abinash Nayak

In the partial fulfilment for the award of the degree of Bachelor of Technology in Computer science and Engineering, Ajay Binay Institute of Technology, Cuttack of Biju Patnaik University of Technology, Odisha in the year 2022-2023 is an authentic work carried out under our guidance and supervision.

The matter embodied in this report has not been submitted to any other university/institute for the award of any degree to the best of our knowledge.

Er. Abhipsa Pattanaik

SUPERVISOR / GUIDE

Prof. Dr. RAJESH KUMAR SAHOO

Head of the Department of Computer science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AJAY BINAY INSTITUTE OF TECHNOLOGY, CUTTACK

CERTIFICATE

This is to certify that the project report entitled “Flappy Bird Using Python” is the work done by:

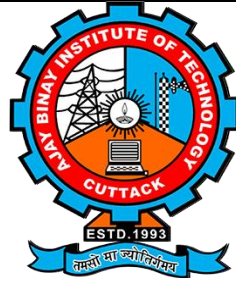
1901206086	Sourav Patnaik
1901206040	Abhijeet Kumar Sahoo
1901206089	Subhrajit Bastia
1901206042	Abinash Nayak

of BTech (Computer science and Engineering) of ABIT, Cuttack under BPUT, Odisha submitted in partial fulfilment for the award of degree.

We are satisfied that they have worked sincerely and with proper care.

Supervisor/Guide
Er. ABHIPSA PATTANAIAK

H.O.D
Prof. Dr. RAJESH KUMAR SAHOO



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING AJAY BINAY INSTITUTE OF TECHNOLOGY, CUTTACK

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mentioning of the people whose constant guidance and encouragement made it possible. I take pleasure in presenting before you, our project, which is result of studied blend of both research and knowledge.

I express our earnest gratitude to Er. ABHIPSA PATTANAIK Department of CSE, our project guide, for her constant support, encouragement, and guidance. I am grateful for her cooperation and valuable suggestions.

I feel to avail myself of this opportunity to express my deep sense of gratitude to Prof. RAJESH KUMAR SAHOO , HOD, Dept. of Computer science and Engineering, for the facilities made available and instructions given to me in accomplishing this project successfully.

Finally, we express our gratitude to all other members who are involved either directly or indirectly for the completion of this project.

1901206086

Sourav Patnaik

1901206040

Abhijeet Kumar Sahoo

1901206089

Subhrajit Bastia

1901206042

Abinash Nayak

CONTENTS

1) INTRODUCTION-----7

2) WHY THIS GAME?-----8

3) WHY USE PYTHON?-----9

4) TECHNOLOGY AND SOFTWARES USED-----10

5) SRS DOCUMENT-----12

6) DFD OF FLAPPY BIRD-----23

7) METHODOLOGY-----24

8) SOURCE CODE-----27

9) GAMEPLAY-----32

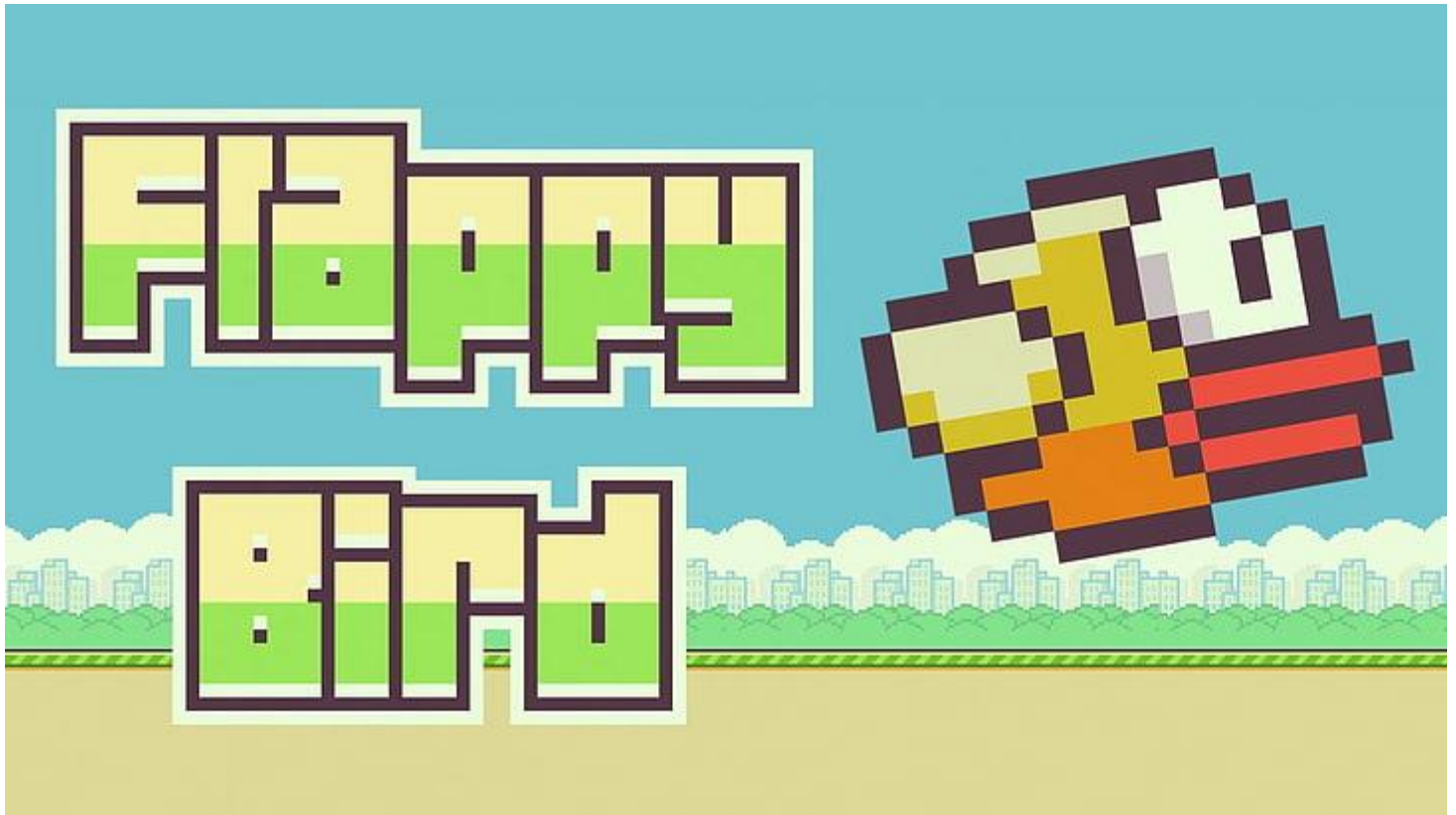
10) TESTING-----35

11) CONCLUSION-----36

12) REFERENCES-----37

INTRODUCTION

Our software project is Flappy Bird. It's a game application. The game was designed and built by Dong Nguyen, a developer who lives in Vietnam. Flappy bird is a side-scroller game where the player controls a bird, attempting to fly between columns of green pipes. The bird will be flying until it collisions with a pipe or it fall on ground. It's a simple game of infinite level type. It's a challenging game for all. Everyone can play this game of every age group. It is addictive as well.



WHY THIS GAME ?

We choose this game because first we are new to this field and we wanted to develop a project which is easy to make and easily accessible to any type of user.

This is modern world and esports is growing in India. We can see many esports player growing and getting a steady income source for example scout , mortal etc. Not only esports streaming career as well. People are choosing games for streaming or esports competition as their jobs. There are many organizations who are giving chances to these people to make name, fame , money like Velocity gaming , 8bit , S8UL etc.

Our main purpose to choose this game is that from early age group to very old age people can access this game and get an idea that you can choose gaming as your career as well. We wanted to establish that everyone can from early age group can be a live streamer or esports player.

This game flappy bird is easy to play and every age group of people can play this game easily.



WHY USE PYTHON?

Thanks to the language's simplicity and coding speed, Python is an excellent choice for prototyping. The effect of your work is visible immediately and it's possible to quickly deliver a playable project to potential investors.

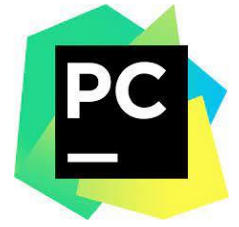
Therefore, Python is also an ideal language to begin your adventure with creating games, regardless of whether you're already fluent in it or just starting out.

Due to the newcomer-friendly syntax of Python, developers can focus on the basics of game programming, not the complexity of the language itself.

The effects, which are quickly visible, allow beginners to get actively involved in the process of creating a project. Advanced developers, on the other hand, will appreciate the fact that they can create something playable using their everyday work tool, and that the process of coding will be, as always, very enjoyable and satisfying.

TECHNOLOGY AND SOFTWARE USED

We used PYTHON programming language to develop our game and PYCHARM Application is used to write the code. We also used PYGAME library to make our game. We also used a converter to convert our python file to exe file so that anyone can access the even those who does not have python installed. We used all these in our pc or desktop to make our game.



Flappy Bird Game using PyGame in Python

Python is one of the most popular high-level programming languages. Python offers huge libraries for different fields like **Artificial Intelligence** (TensorFlow, PyTorch), **Machine Learning** (Pandas, NumPy, Matplotlib), and **Game Development** (Pyglet, PyGame). We can also consider Python as the next-generation programming language as it shows its presence actively in every emerging field of Computer Science.

Let us briefly understand the Python **PyGame** library.

Understanding the PyGame library

The **PyGame** library is a cross-platform set of Python modules utilized to develop video games. **PyGame** mainly comprises computer graphics and sound libraries designed to be utilized with the Python programming language. **Pete Shinnars** officially wrote this library to replace **PySDL**. It is suitable to develop client-side applications that can be potentially wrapped in a standalone executable.

How to install the PyGame library?

The **PyGame** library can be installed using the **PIP** installer by typing the following command in a command prompt or terminal.

Syntax:

1. `# installing the PyGame library`
2. `$ pip install pygame`

Once the installation is complete, we can verify whether the **pygame** library is installed properly or not by creating a new python program file and importing the **pygame** module.

The following is the snippet of code illustrating the same.

File: verify.py

1. `import pygame`

Now, let us save the file and run the following command in a command prompt or terminal.

Syntax:

1. `$ python verify.py`

The library has been installed successfully if the program does not return any importing error. In case any exception is raised, try reinstalling the library and consider checking their official documentation.

PYCHARM

PyCharm is an integrated development environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

It is cross-platform, working on Microsoft Windows, macOS and Linux. PyCharm has a Professional Edition, released under a proprietary license and a Community Edition released under the Apache License. PyCharm Community Edition is less extensive than the Professional Edition.

Features

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python code refactoring: including rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask
- Integrated Python debugger
- Integrated unit testing, with line-by-line coverage
- Google App Engine Python development
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with changelists and merge
- Scientific tools integration: integrates with IPython Notebook, has an interactive Python console, and supports Anaconda as well as multiple scientific packages including Matplotlib and NumPy.

Auto PY to EXE

A .py to .exe converter using a simple graphical interface and [PyInstaller](#) in Python.

You can install this :

```
pip install auto-py-to-exe
```

Then to run it, execute the following in the terminal:

```
$ auto-py-to-exe
```

Software Requirements Specification for Flappy Bird

Version 1.0

Prepared by

Group 11

Abhijeet Kumar Sahoo	1901206040	abhijeetkumarsahoo381@gmail.com
Sourav Patnaik	1901206086	souravpatnaik59396@gmail.com
Subhrajit Bastia	1901206089	subhrajitbastia2002@gmail.com
Abinash Nayak	1901206042	nayakabinash933@gmail.com

Guide: *Abhipsa Pattanaik*
Course: B.tech (CSE)
Sem: 7th
Date: 20/12/2022

Contents

CONTENTS.....13

1 INTRODUCTION14

1.1 DOCUMENT PURPOSE14

1.2 PRODUCT SCOPE144

1.3 INTENDED AUDIENCE AND DOCUMENT OVERVIEW.....144

2 OVERALL DESCRIPTION15

2.1 PRODUCT OVERVIEW15

2.2 DESIGN AND IMPLEMENTATION CONSTRAINTS15

2.3 ASSUMPTIONS AND DEPENDENCIES15

3 FUNCTIONALREQUIREMENTS16

3.1 ACTIVITY MODEL17

3.2 DATA FLOW DIAGRAM18

4 INTERFACE REQUIREMENTS19

5 PERFORMANCE REQUIREMENTS20

6 NON-FUNCTIONAL REQUIREMENTS.....21

7 OPERATIONALREQUIREMENTS22

8 APPENDICES.....23

8.1 DEFINITIONS, ACRONYMS AND ABBREVIATIONS.....23

8.2 REFERENCES AND ACKNOWLEDGMENTS23

Revision History

Name	Date	Reason For Changes	Version
Sourav Patnaik	19/12/2022	SRS document is made.	1.0
Sourav Patnaik	29/03/2023	SRS document was added to the report	1.0

1 Introduction

1.1 Purpose

This is a software requirements specification document about our project “Flappy Bird”, which provides a complete description of all the properties that will be implemented and a complete definition of system attributes. Specialties of the system and functionalities of the project will be defined, explained and demonstrated by using some models.

The main purpose of the project is to create an entertaining and a unique game which runs on the PC platforms. This document’s audience is developers, testers, end users.

In short, the purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements.

1.2 Product Scope

The project’s name is “Flappy Bird”. The scope of this project is to create a unique which are controlled by different kinds of mechanisms at the same time. The aim of this project is to create an entertaining and stress free game. This project consists of one basic mode which is single player mode.

The game is a single controller where the player controls a bird, attempting to fly between rows of pipes without coming into contact them. If the player touches the pipes, it ends the game. The bird briefly flaps upward each time the player presses the spacebar or upper arrow key button ; if not pressed, the bird can also fall due to lack of acceleration by user .

1.3 Intended Audience and Document Overview

This document is designed to be read and looked by our group members and other students , our guide , instructor ,teachers ,faculties and other staff members of our college.

Also this project or the game can be accessed by every age group like children adults and old age people.

2 Overall Description

2.1 Product Overview

This game inherits "Flappy Bird" Operation Simple, the rhythm is crisp, and we also add some innovative elements. Want to make computer workers busy, enjoy the fun of the game. The interface of the game strives for beauty, pleasing to the eye, music effect also strive to be realistic, fascinating.

Flappy Bird is an endless game that involves a bird that the player can control. The player has to save the bird from colliding with the hurdles like pipes. Each time the bird passes through the pipes, the score gets incremented by one. The game ends when the bird collides with the pipes or falls down due to gravity.

2.2 Design and Implementation Constraints

PC version of the "bird" and may be mobile phone version of the "bird" function, mainly for the user base is different.

2.3 Assumptions and Dependencies

- The time constraint for developing this game is about 7 weeks.
- Different versions may be made.

3 Functional Requirements

- **2D Animation**

Animation is a complex subject in game programming. Animation is rapid display of sequence of images which creates an illusion of movement. Python games are expected to run on multiple operating systems with different hardware specifications. Threads give the most accurate timing solutions.

- **Objective Selection**

We create a bird object which is flying until any collision occurred and the bird is flying in the wall objectives which are begin from top and bottom of the screen.

- **Moving Pipes**

The pipes moving on and it will come randomly in size and distances. The bird is flying in the middle of the pipes.

- **Collision Detection**

When the bird touch the anywhere of a pipe it cause a collision. Collision detection is one of important task of the game. If the bird touch any wall (pipes) the game will end.

- **Moving Background**

The picture used as background image is moving on analogously. We used two same image which are coming one after another regularly.

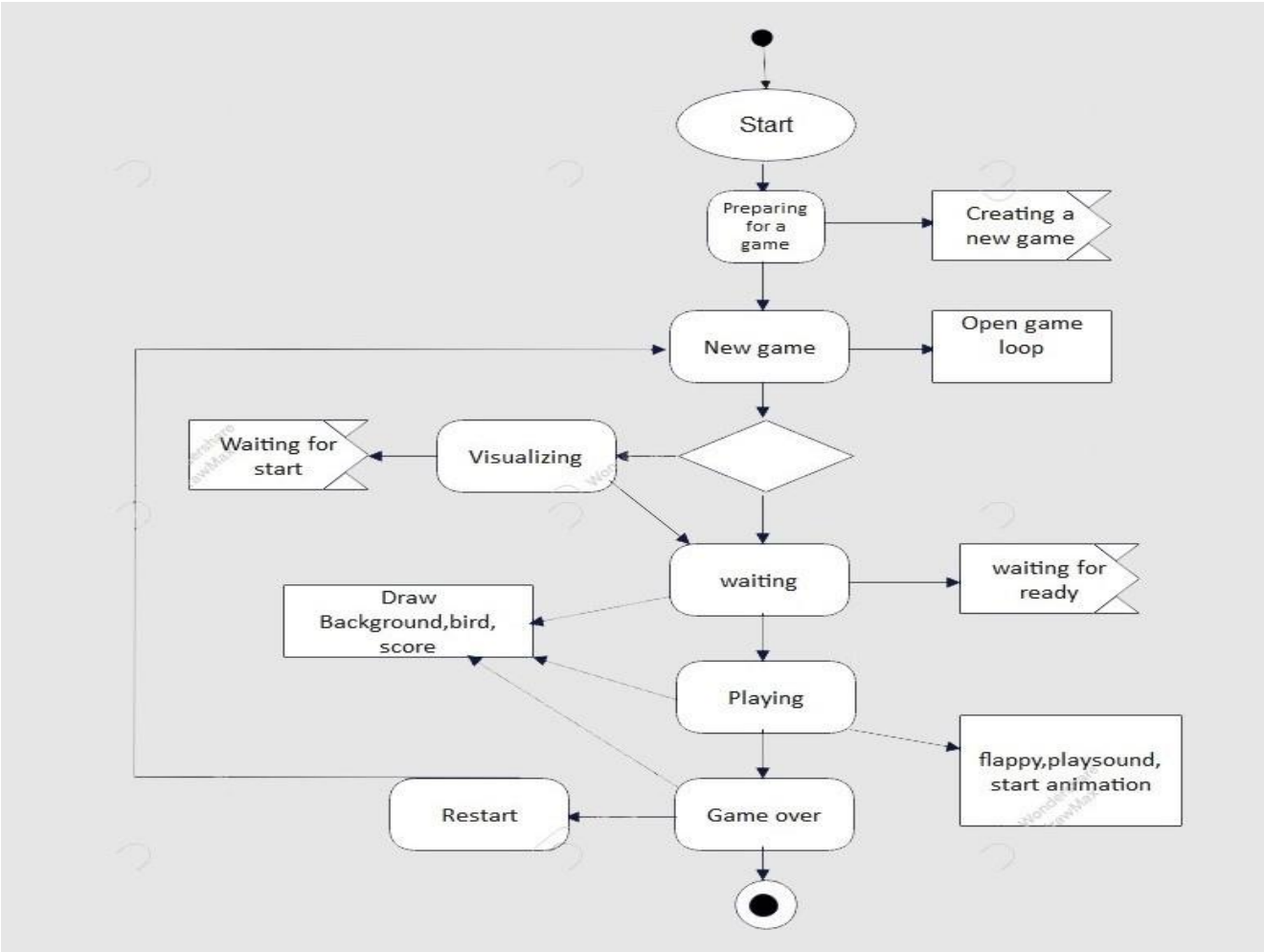
- **Score Counting**

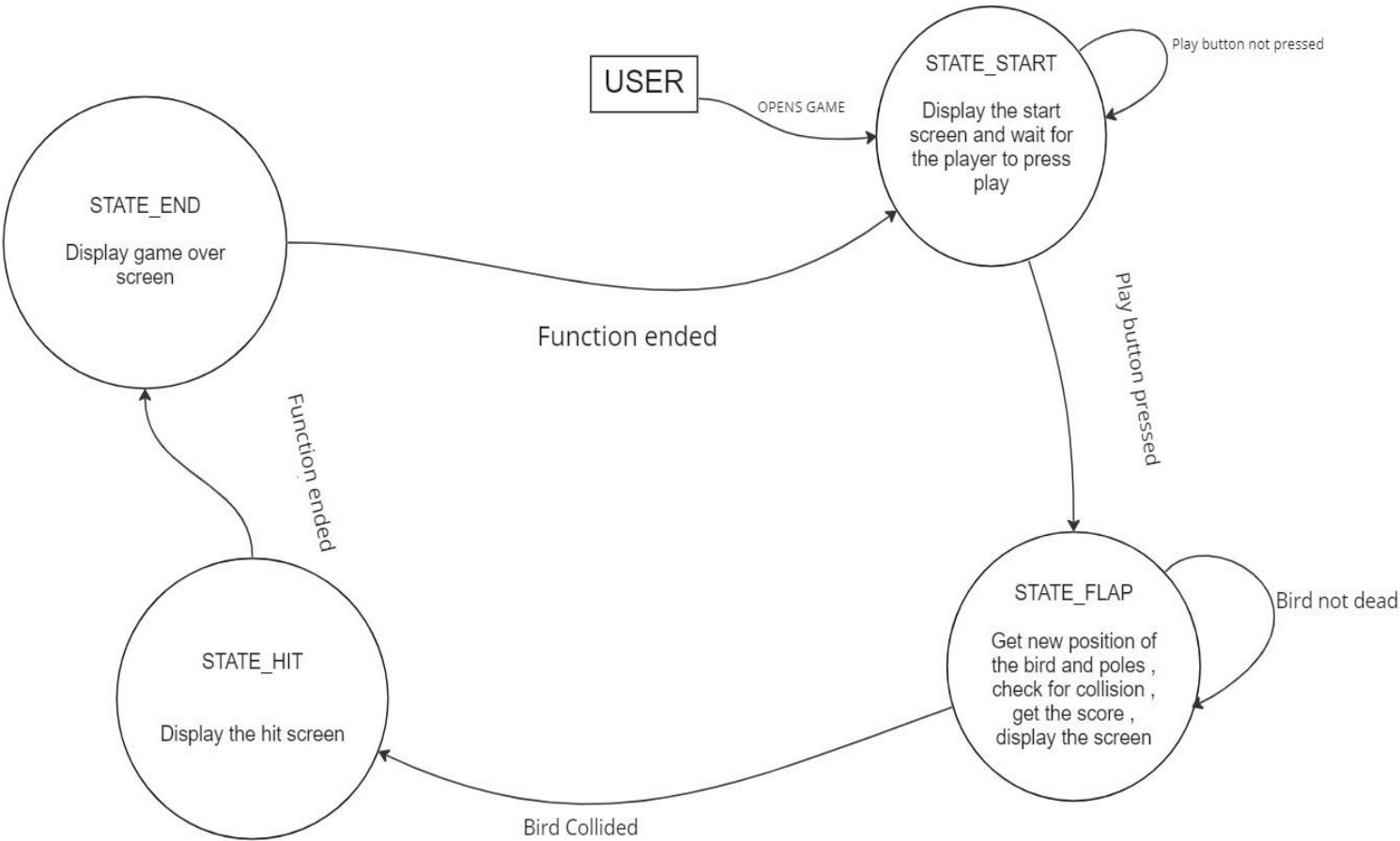
Score counting is the interesting for user. By the score the player knows his/her performance. If the bird cross a pipe without collision or not fall in ground his/her score increment one.

- **Sound**

Sound is given to the game. Background music , mouse click sound ,collision sound, score count sound will be given.

3.1 Activity Model :-





DFD FOR FLAPPY BIRD GAME

4 Interface Requirements

Main interface-

Start screen of the game which will allow the user to start the game.

Game interface-

Game screen which shows process of the game like the bird flying , background , pipes, bird colliding ,etc.

5 Performance Requirements

Accuracy and flexibility :-

The game requires a single press on the spacebar or upper arrow key button, must be in real-time response to the specified height, and the bird has falling speed, and the column must have been circulating, and finally make the bird flexible and coherent to shuttle between the pillars.

Time characteristics :-

This includes the time characteristics of the bird flying upward, the drop, and the time of the column cycle.

input and output requirements :-

Input: Click the left mouse button to control the bird flying upward. Tap the music icon to control sound effects such as background music and flight collisions.

Output: During the game, each time the bird crosses through a pillar it shows the score plus one; After the game, output "game Over" and show the current score (score).

6 Non-functional Requirements

Functionality: effectively implement the relevant content described in functional requirements.

Reliability: Require the system to operate for a long time, and support multi-user simultaneous access;

Ease of use: User-friendly interface and simple operation method to ensure the player's easy to use

Security: To ensure that users in the use of software process data security, communication information security, transmission of file security.

High-performance: to meet the smooth game needs, no obvious lag in the game process, delay the number of seconds below; file transfer is fast.

Maintainability: Can meet the system administrator's need of system maintenance, can realize the software developer's further maintenance to the software;

Extensibility: To meet the needs of software developers to further expand the functionality of the software;

Testability: When needed, it is possible to control the output of internal critical information through a configuration file, and the output target can also be configured through a configuration file

Operating Environment Regulations : -

Operating system: Microsoft Windows 8 , Microsoft Windows 10 , Microsoft Windows 11.

Development environment :-

Development environment: Pycharm , Pygame libraries and so on.

8.1 Definitions, Acronyms and Abbreviations

- *PC – Personal Computer*
- SRS - Software Requirements Specification.
- MS- Microsoft

8.2 References and Acknowledgments

- https://en.wikipedia.org/wiki/Flappy_Bird
-

METHODOLOGY

- Firstly, we will import the required modules.
- Then the classes and objects are added for background ,bird, pipes, main menu and game over screen.
- We add relevant game logic for smooth play and in accordance to screen and system settings
- We add the Sprite Animation to the Game.
- We will then add some physics to the Game.
- We will then add a score counter.
- We will last add functions for game over and reset the game.
- Once everything is in check the user can execute the code and run the game.
- We also made an exe file so that everyone can play the game without any restrictions.

GAME LOGIC :-

It is true that Flappy Bird is one of the legendary games we used to play, or perhaps some still play the game until today. Played by hundreds of thousands of players globally, the game became a hot trend a few decades ago.

The objective of Flappy Bird is simple: attempting to fly the bird between pipes without hitting them. The longer the bird survives the moving pipes, the more points players will gain. From a gamer view, this game might seem simple. On the other hand, from a coder perspective, Flappy Bird is indeed an interesting topic to discuss.

A Little Background

Although you might already know what a Flappy Bird program looks like, please check out our program to get an image as to what we'll cover in this essay. The main principles of every other flappy bird game are identical. For instance, in the original Flappy Bird, the green pipes fills the whole screen but are divided into half same is with our program. Also, while the original game has a portrait orientation and is more dedicated to mobile users, but our game is more dedicated to pc users and for mobile it can be developed.

Since we'll be discussing logics and algorithms, not syntaxes, it does not matter which programming language you are using but we made our game using python programming language and later it is converted into executable file

Now that you have the big image, we're ready to uncover each points in detail. There are four main components in this game: a bird, several pipes, background, sound etc.

First Step Towards Success

The first step towards creating a successful Flappy Bird is to create a bird character by either drawing it manually with code or uploading an image of a bird to the file. As responsible programmers, we need to set the size of the bird to a rational size, not too big nor too tiny.

Next, we need to define a y variable to store the current vertical position of the bird. An x variable is not necessary since the bird will not be moving left or right — the pipes will! Check whether the y variable is connected to the bird image by changing the value of the y variable. If the bird moved upwards/downwards, we are ready to move on to the next step: automating the bird movement.

To do this, three important elements need to be implemented to our program: gravity, user interaction, and limit. It is easy to “create” gravity. We just need to continuously add the y variable we have defined earlier to some logical number, thus seemingly bringing the bird down to ground.

After gravity, we’ll discuss user interaction. The only method to have user interacting with the bird is to use a conditional statement; if users do something, then move the bird upwards by reducing the y variable to some value that is greater than the amount of gravity. Decide whether you want users to interact by touching their computer screen or pressing a certain key on their keyboard (or both).

Finally, we’ll set limits to the bird. Obviously, we do not want the bird, or part of the bird, to be seen off-screen because of flying too high or falling too low. To limit the bird’s movements, use another conditional statement to ensure that the bird’s y variable is more than 0 and less than the maximum height of your program screen.

And there you have it, a fully interactive bird!

Challenging the Limits

Merely clicking to see the bird fly might seem dull after some time, though. In order not to make it boring, our game should challenge player’s focus and skill. Hence, let’s add obstacles to our program. There is one main obstacle our Flappy Bird Game pipes. We’ll cover the pipes first.

We will need an image of a pipe. Again, getting the image can be done by either drawing it or uploading an existing image. When playing the game, notice that the bird has probably passed through thousands of pipes. However, the fact is that only two pipes are required. Both of the pipes are put on a cycle , which was mentioned earlier, to

illustrate that effect. Obviously, we will need some variables to store important values, such as the x position, y position, and height of both pipes. Then, animate the pipes and put them on a cycle. Now here is where the addition comes. When a pipe reaches the top left part of the screen, instead of directly changing the x variable equal to screen width, we need to randomize the y position of that pipe, adjust the height accordingly, then change the x variable to move the pipe. Doing this will allow players to think that each pipe the bird passes is unique and thus improving overall game experience.

Death is Inevitable

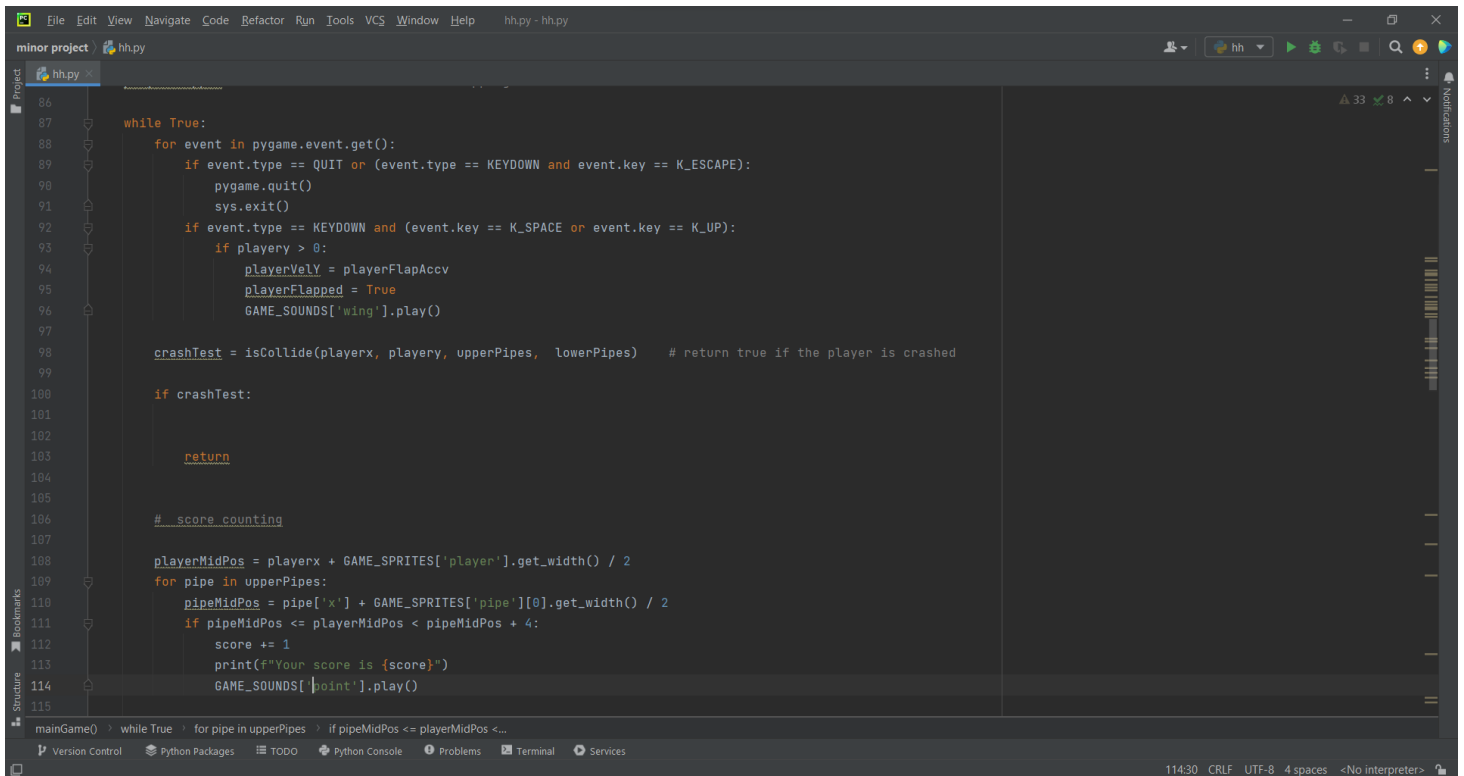
To every beginning, there is always an end. We need to somehow end our game because no one wants to play a never ending game. First, define a new variable to check whether game is over. Set the initial value of that variable to false.

Next, we need to detect whether the bird collides with a pipe. A function is required to stop the game. Any collision should trigger that function. Each collision detection varies by programming languages, but the logic is the same: if the bird hits an obstacle, set the game over variable value to true. End the game with something fancy like a pop up or whatever you like.

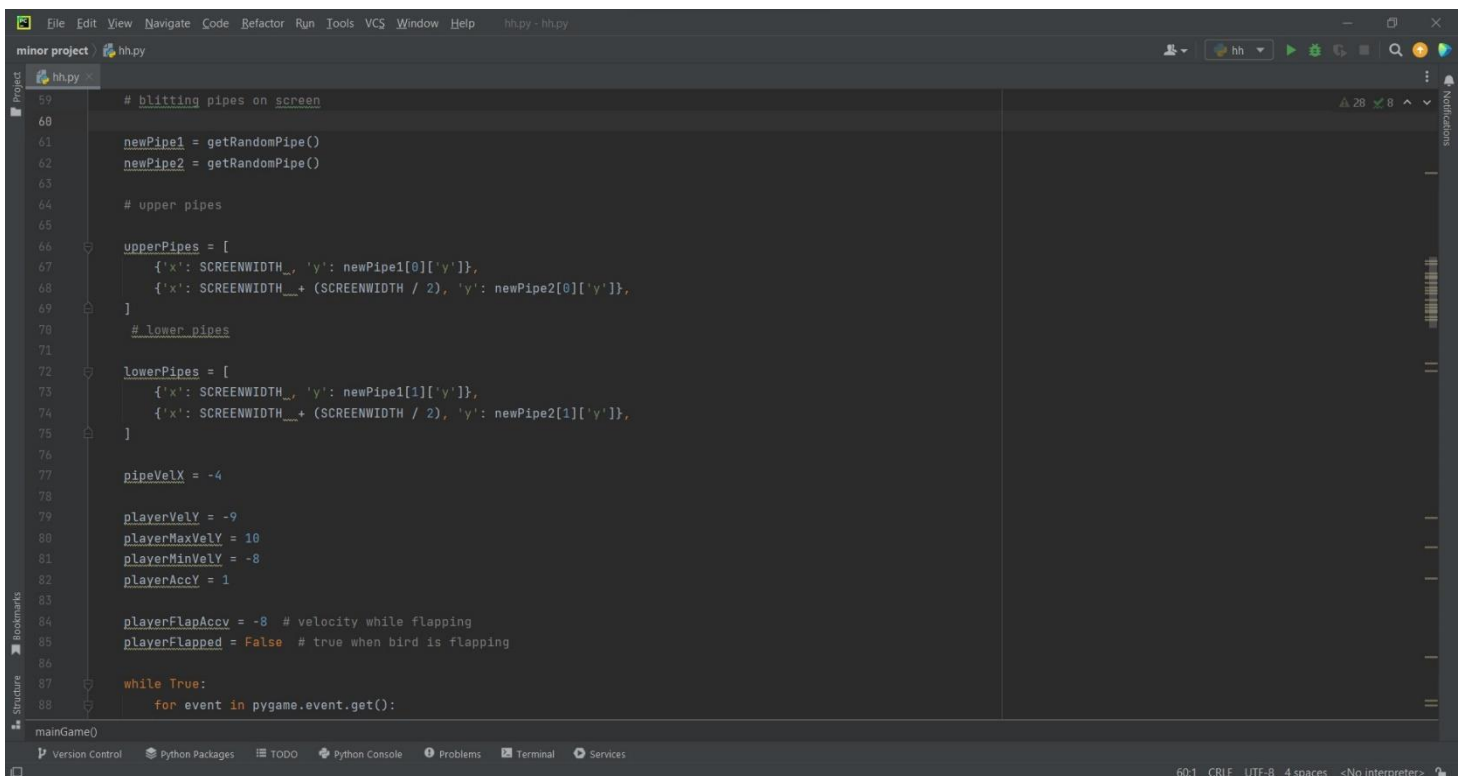
Final Touch

Finally, we will add some decorations specifically a background. Draw roads, bushes, cars, or anything that fits our theme or download and upload an image. Animating some of the decorations or the background is highly recommended so that our Flappy Bird environment becomes more lively. This part is not necessary but it can immensely improve our game appearance. We also added sound and score to feel more lively and competitive. Now that we have understood the process required to create your own Flappy Bird, why create one? Perhaps we might learn something interesting along the way that can further boost our knowledge.

SOURCE CODE

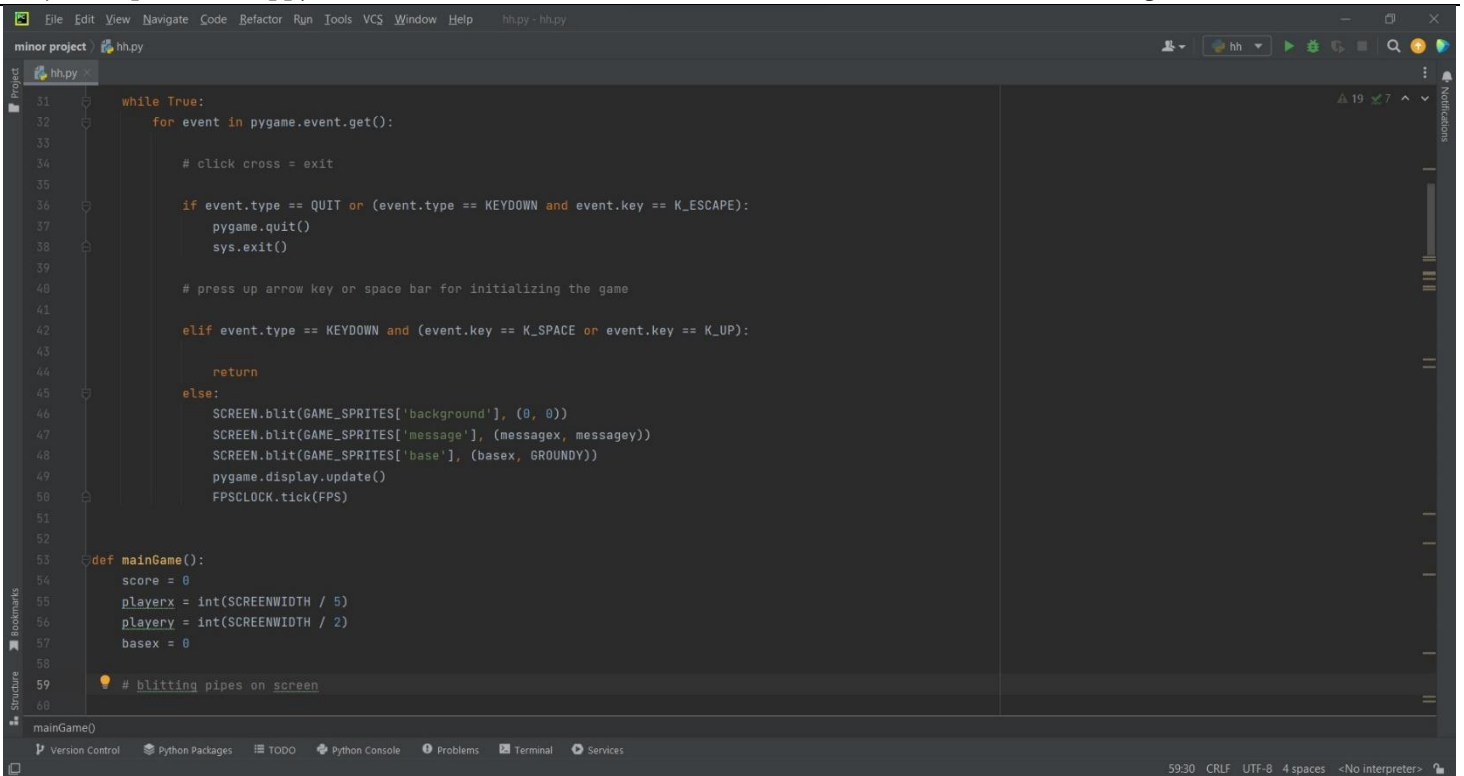


```
86
87 while True:
88     for event in pygame.event.get():
89         if event.type == QUIT or (event.type == KEYDOWN and event.key == K_ESCAPE):
90             pygame.quit()
91             sys.exit()
92         if event.type == KEYDOWN and (event.key == K_SPACE or event.key == K_UP):
93             if playery > 0:
94                 playerVelY = playerFlapAccv
95                 playerFlapped = True
96                 GAME_SOUNDS['wing'].play()
97
98     crashTest = isCollide(playerx, playery, upperPipes, lowerPipes) # return true if the player is crashed
99
100     if crashTest:
101
102
103         return
104
105     # _score counting
106
107     playerMidPos = playerx + GAME_SPRITES['player'].get_width() / 2
108     for pipe in upperPipes:
109         pipeMidPos = pipe['x'] + GAME_SPRITES['pipe'][0].get_width() / 2
110         if pipeMidPos <= playerMidPos < pipeMidPos + 4:
111             score += 1
112             print(f"Your score is {score}")
113             GAME_SOUNDS['point'].play()
114
115
```

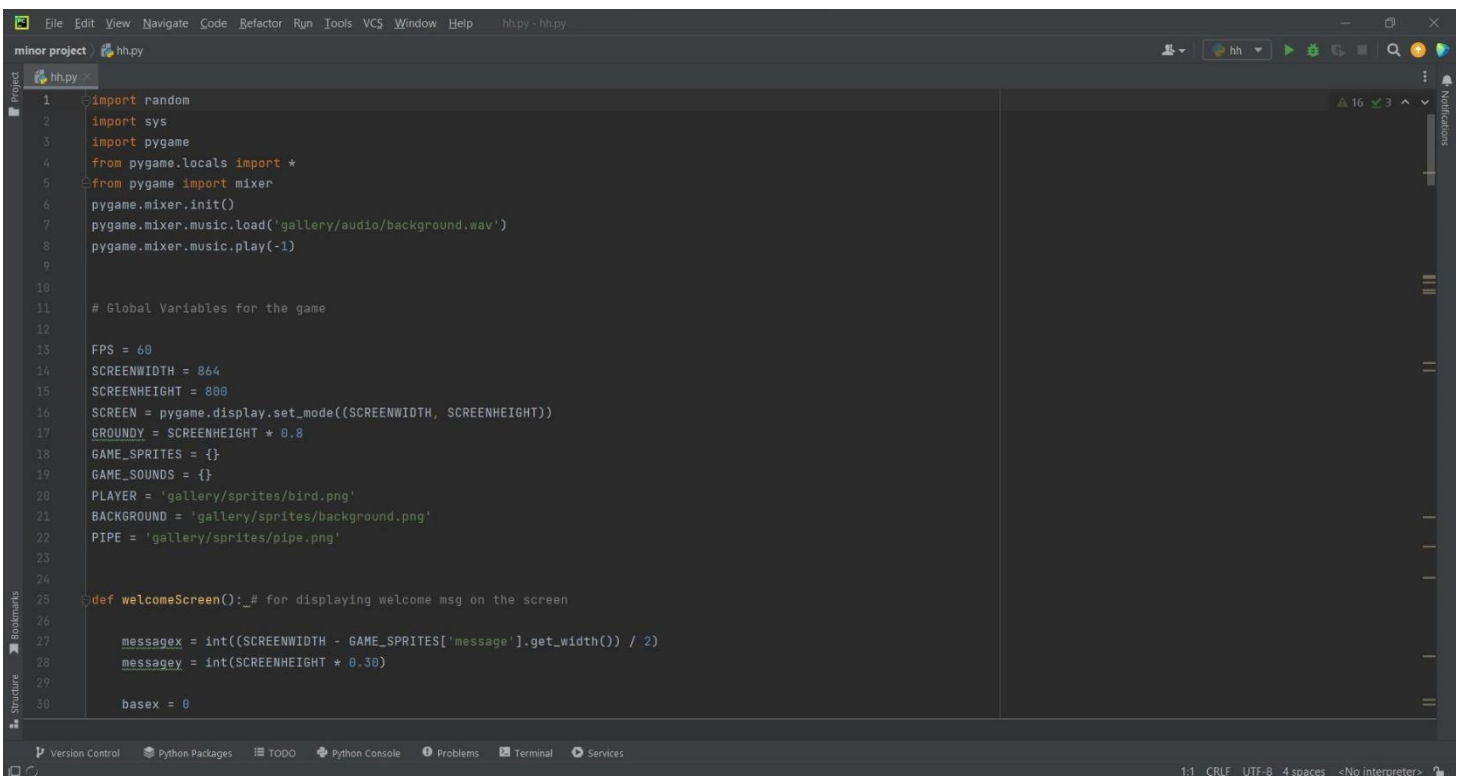


```
59 # blitting pipes on screen
60
61 newPipe1 = getRandomPipe()
62 newPipe2 = getRandomPipe()
63
64 # upper pipes
65
66 upperPipes = [
67     {'x': SCREENWIDTH_, 'y': newPipe1[0]['y']},
68     {'x': SCREENWIDTH_ + (SCREENWIDTH / 2), 'y': newPipe2[0]['y']},
69 ]
70 # lower pipes
71
72 lowerPipes = [
73     {'x': SCREENWIDTH_, 'y': newPipe1[1]['y']},
74     {'x': SCREENWIDTH_ + (SCREENWIDTH / 2), 'y': newPipe2[1]['y']},
75 ]
76
77 pipeVelX = -4
78
79 playerVelY = -9
80 playerMaxVelY = 10
81 playerMinVelY = -8
82 playerAccy = 1
83
84 playerFlapAccv = -8 # velocity while flapping
85 playerFlapped = False # true when bird is flapping
86
87 while True:
88     for event in pygame.event.get():

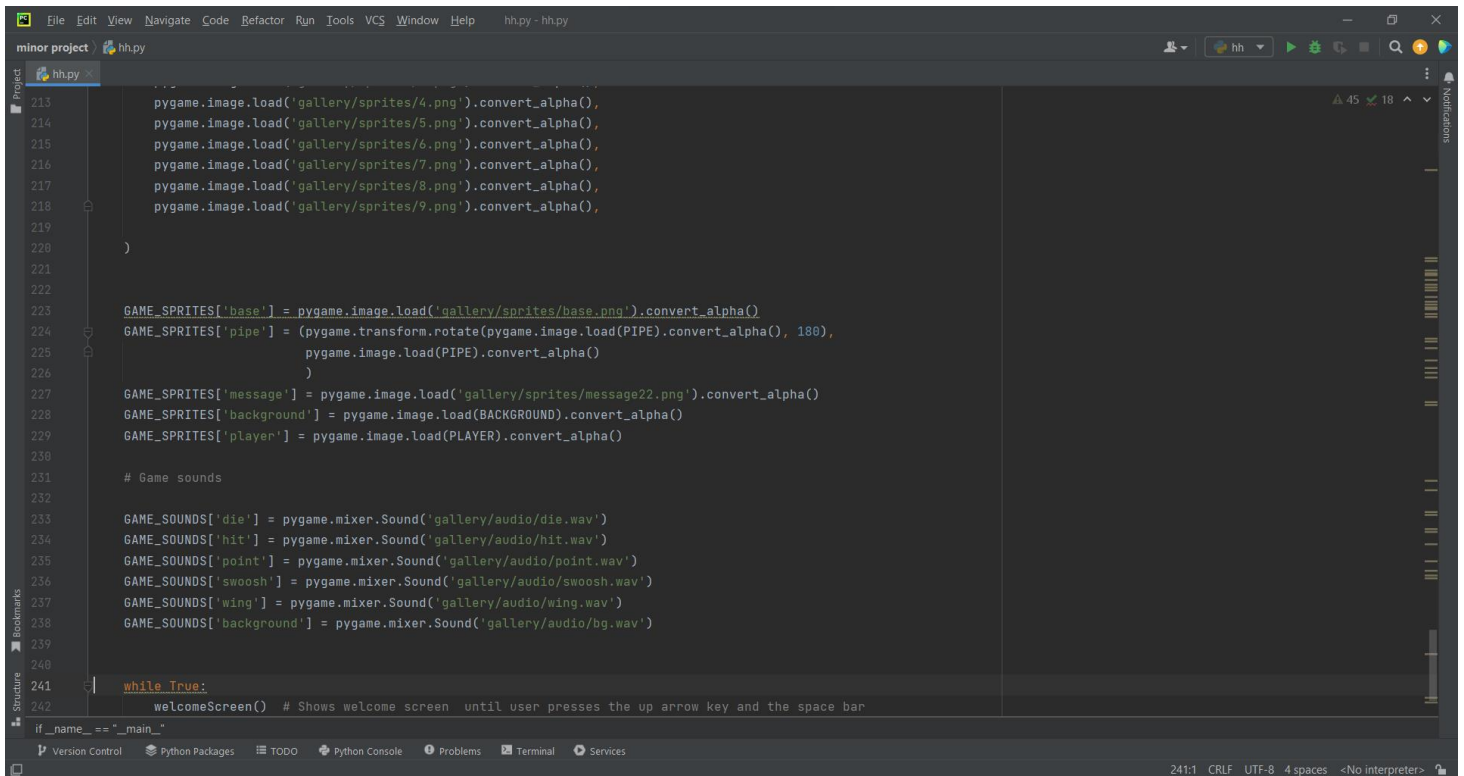
```



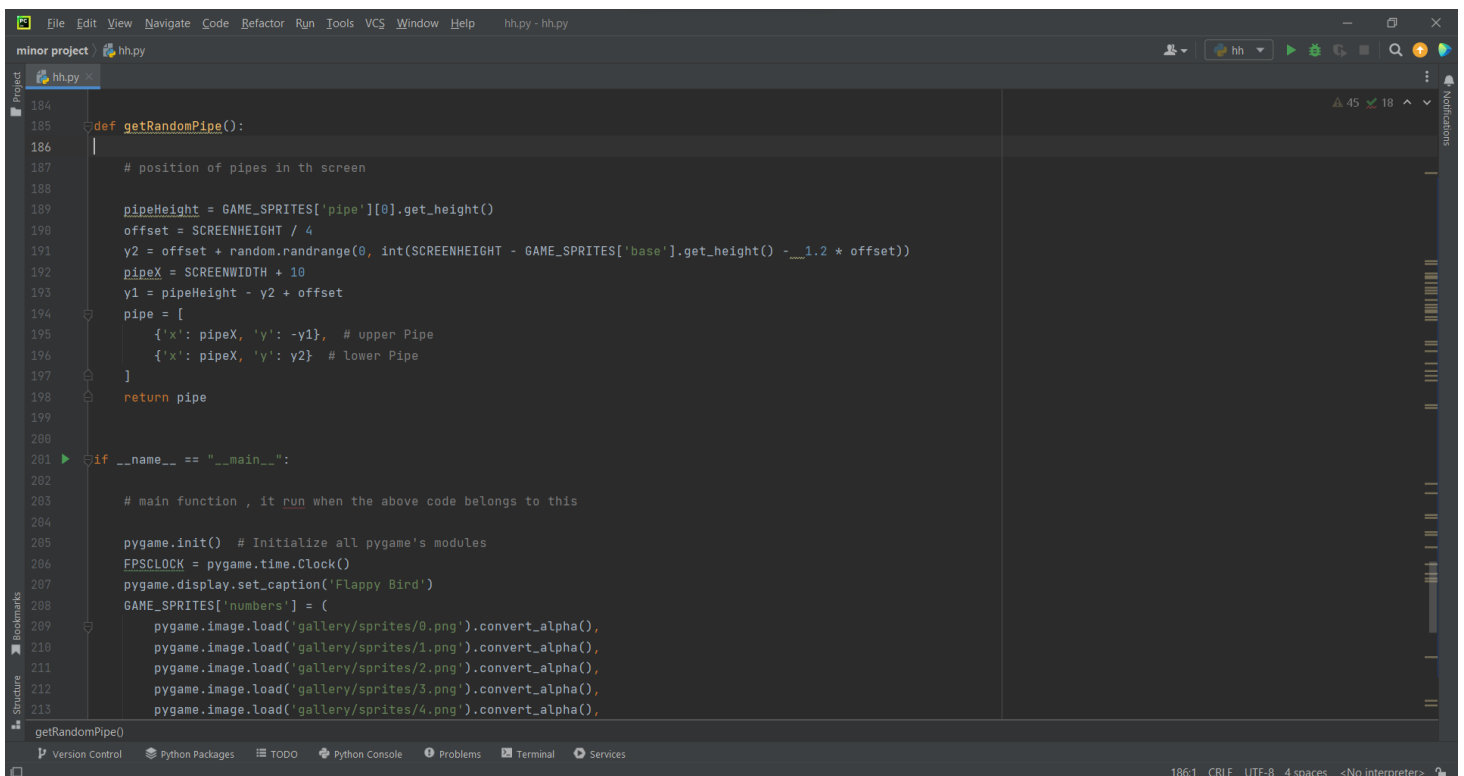
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hh.py - hh.py
minor project hh.py
31 while True:
32     for event in pygame.event.get():
33
34         # click cross = exit
35
36         if event.type == QUIT or (event.type == KEYDOWN and event.key == K_ESCAPE):
37             pygame.quit()
38             sys.exit()
39
40         # press up arrow key or space bar for initializing the game
41
42         elif event.type == KEYDOWN and (event.key == K_SPACE or event.key == K_UP):
43
44             return
45         else:
46             SCREEN.blit(GAME_SPRITES['background'], (0, 0))
47             SCREEN.blit(GAME_SPRITES['message'], (messagex, messagey))
48             SCREEN.blit(GAME_SPRITES['base'], (basex, GROUNDY))
49             pygame.display.update()
50             FPSLOCK.tick(FPS)
51
52
53 def mainGame():
54     score = 0
55     playerx = int(SCREENWIDTH / 5)
56     playery = int(SCREENWIDTH / 2)
57     basex = 0
58
59     # blitting pipes on screen
60
mainGame()
Version Control Python Packages TODO Python Console Problems Terminal Services
59:30 CRLF UTF-8 4 spaces <No interpreter>
```



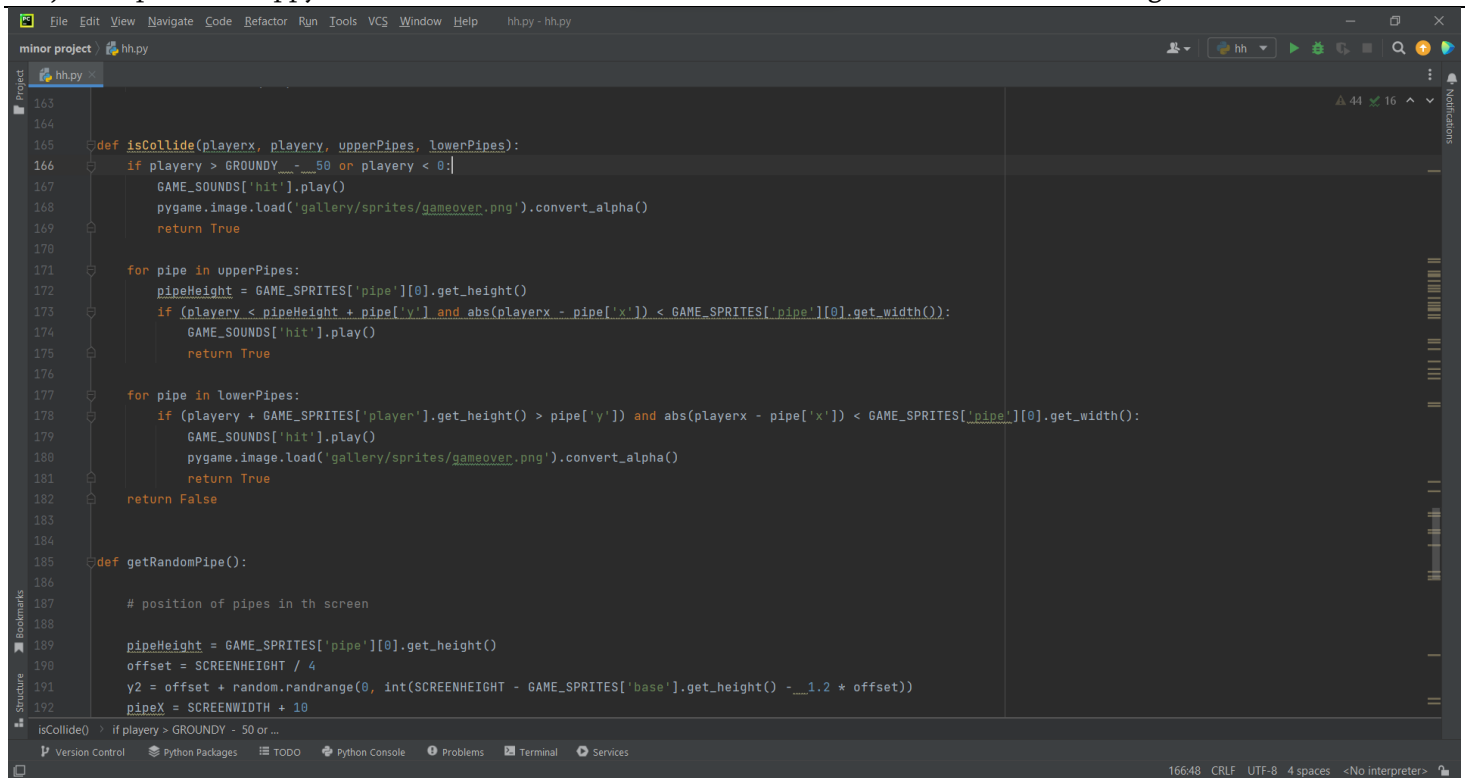
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hh.py - hh.py
minor project hh.py
1 import random
2 import sys
3 import pygame
4 from pygame.locals import *
5 from pygame import mixer
6 pygame.mixer.init()
7 pygame.mixer.music.load('gallery/audio/background.wav')
8 pygame.mixer.music.play(-1)
9
10
11 # Global Variables for the game
12
13 FPS = 60
14 SCREENWIDTH = 864
15 SCREENHEIGHT = 800
16 SCREEN = pygame.display.set_mode((SCREENWIDTH, SCREENHEIGHT))
17 GROUNDY = SCREENHEIGHT * 0.8
18 GAME_SPRITES = {}
19 GAME_SOUNDS = {}
20 PLAYER = 'gallery/sprites/bird.png'
21 BACKGROUND = 'gallery/sprites/background.png'
22 PIPE = 'gallery/sprites/pipe.png'
23
24
25 def welcomeScreen(): # for displaying welcome msg on the screen
26
27     messagex = int((SCREENWIDTH - GAME_SPRITES['message'].get_width()) / 2)
28     messagey = int(SCREENHEIGHT * 0.30)
29
30     basex = 0
Version Control Python Packages TODO Python Console Problems Terminal Services
1:1 CRLF UTF-8 4 spaces <No interpreter>
```



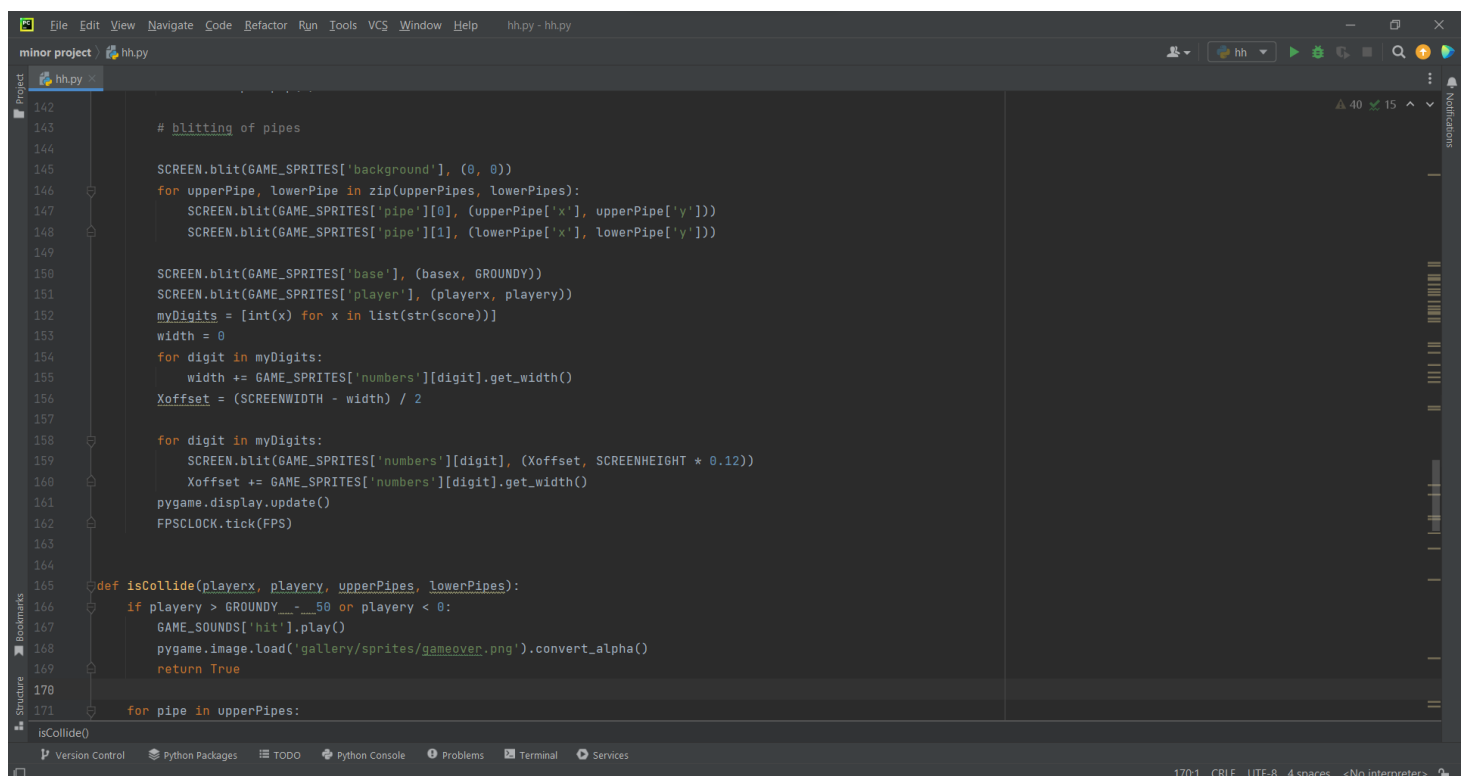
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hh.py - hh.py
minor project hh.py
213 pygame.image.load('gallery/sprites/4.png').convert_alpha(),
214 pygame.image.load('gallery/sprites/5.png').convert_alpha(),
215 pygame.image.load('gallery/sprites/6.png').convert_alpha(),
216 pygame.image.load('gallery/sprites/7.png').convert_alpha(),
217 pygame.image.load('gallery/sprites/8.png').convert_alpha(),
218 pygame.image.load('gallery/sprites/9.png').convert_alpha(),
219
220 )
221
222
223 GAME_SPRITES['base'] = pygame.image.load('gallery/sprites/base.png').convert_alpha()
224 GAME_SPRITES['pipe'] = (pygame.transform.rotate(pygame.image.load(PIPE).convert_alpha(), 180),
225                         pygame.image.load(PIPE).convert_alpha()
226                     )
227
228 GAME_SPRITES['message'] = pygame.image.load('gallery/sprites/message22.png').convert_alpha()
229 GAME_SPRITES['background'] = pygame.image.load(BACKGROUND).convert_alpha()
230 GAME_SPRITES['player'] = pygame.image.load(PLAYER).convert_alpha()
231
232 # Game sounds
233
234 GAME_SOUNDS['die'] = pygame.mixer.Sound('gallery/audio/die.wav')
235 GAME_SOUNDS['hit'] = pygame.mixer.Sound('gallery/audio/hit.wav')
236 GAME_SOUNDS['point'] = pygame.mixer.Sound('gallery/audio/point.wav')
237 GAME_SOUNDS['swoosh'] = pygame.mixer.Sound('gallery/audio/swoosh.wav')
238 GAME_SOUNDS['wing'] = pygame.mixer.Sound('gallery/audio/wing.wav')
239 GAME_SOUNDS['background'] = pygame.mixer.Sound('gallery/audio/bg.wav')
240
241 while True:
242     welcomeScreen() # Shows welcome screen until user presses the up arrow key and the space bar
243
244 if __name__ == "__main__":
245     welcomeScreen()
246
247 Version Control Python Packages TODO Python Console Problems Terminal Services
241:1 CRLF UTF-8 4 spaces <No interpreter>
```



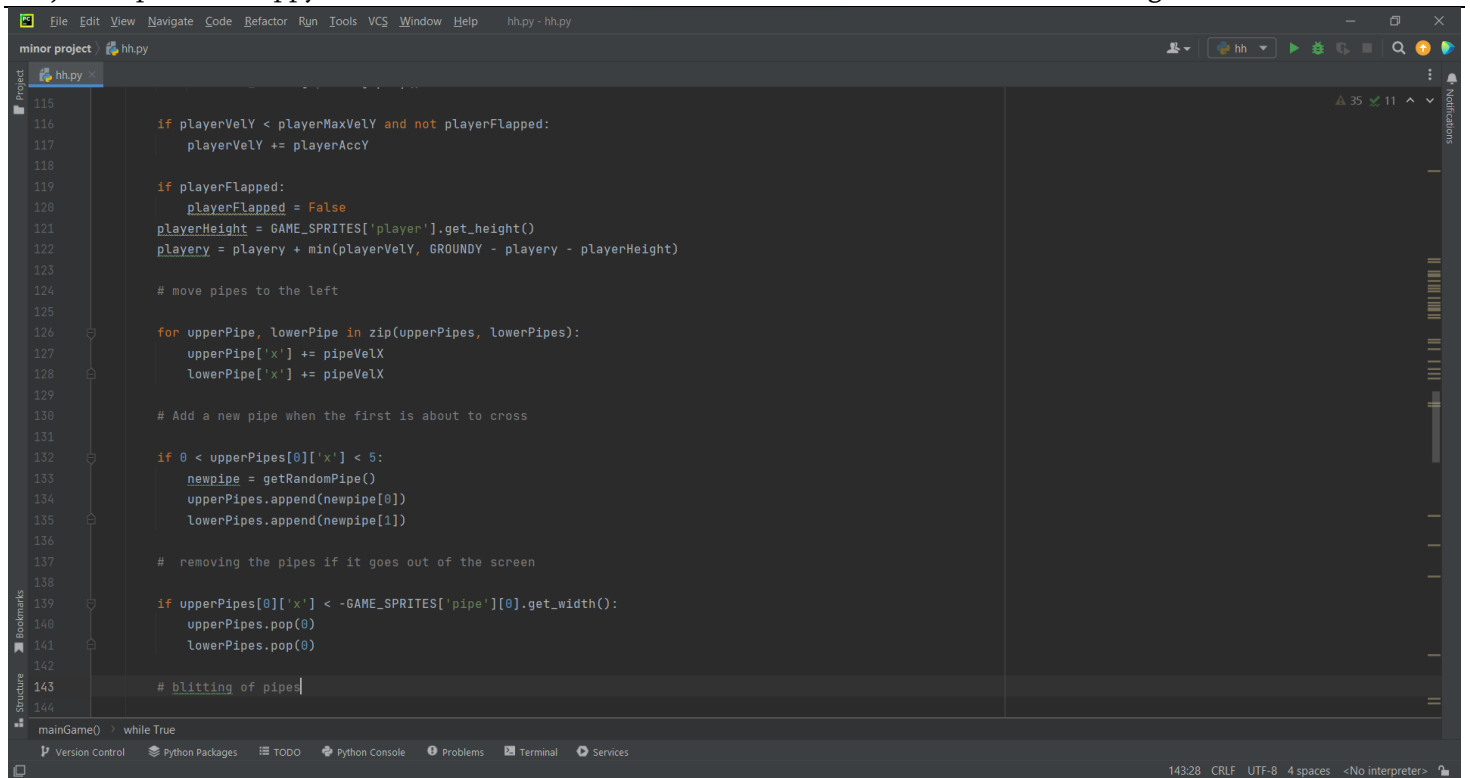
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hh.py - hh.py
minor project hh.py
184
185 def getRandomPipe():
186     |
187     # position of pipes in th screen
188
189     pipeHeight = GAME_SPRITES['pipe'][0].get_height()
190     offset = SCREENHEIGHT / 4
191     y2 = offset + random.randrange(0, int(SCREENHEIGHT - GAME_SPRITES['base'].get_height() - 1.2 * offset))
192     pipeX = SCREENWIDTH + 10
193     y1 = pipeHeight - y2 + offset
194     pipe = [
195         {'x': pipeX, 'y': -y1}, # upper Pipe
196         {'x': pipeX, 'y': y2} # lower Pipe
197     ]
198     return pipe
199
200
201 if __name__ == "__main__":
202     |
203     # main function , it run when the above code belongs to this
204
205     pygame.init() # Initialize all pygame's modules
206     FPSLOCK = pygame.time.Clock()
207     pygame.display.set_caption('Flappy Bird')
208     GAME_SPRITES['numbers'] = (
209         pygame.image.load('gallery/sprites/0.png').convert_alpha(),
210         pygame.image.load('gallery/sprites/1.png').convert_alpha(),
211         pygame.image.load('gallery/sprites/2.png').convert_alpha(),
212         pygame.image.load('gallery/sprites/3.png').convert_alpha(),
213         pygame.image.load('gallery/sprites/4.png').convert_alpha(),
214
215     )
216
217     getRandPipe()
218
219 Version Control Python Packages TODO Python Console Problems Terminal Services
186:1 CRLF UTF-8 4 spaces <No interpreter>
```



```
163
164
165 def isCollide(playerx, playery, upperPipes, lowerPipes):
166     if playery > GROUNDY - 50 or playery < 0:
167         GAME_SOUNDS['hit'].play()
168         pygame.image.load('gallery/sprites/gameover.png').convert_alpha()
169         return True
170
171     for pipe in upperPipes:
172         pipeHeight = GAME_SPRITES['pipe'][0].get_height()
173         if (playery < pipeHeight + pipe['y'] and abs(playerx - pipe['x']) < GAME_SPRITES['pipe'][0].get_width()):
174             GAME_SOUNDS['hit'].play()
175             return True
176
177     for pipe in lowerPipes:
178         if (playery + GAME_SPRITES['player'].get_height() > pipe['y'] and abs(playerx - pipe['x']) < GAME_SPRITES['pipe'][0].get_width()):
179             GAME_SOUNDS['hit'].play()
180             pygame.image.load('gallery/sprites/gameover.png').convert_alpha()
181             return True
182     return False
183
184
185 def getRandomPipe():
186
187     # position of pipes in th screen
188
189     pipeHeight = GAME_SPRITES['pipe'][0].get_height()
190     offset = SCREENHEIGHT / 4
191     y2 = offset + random.randrange(0, int(SCREENHEIGHT - GAME_SPRITES['base'].get_height() - 1.2 * offset))
192     pipeX = SCREENWIDTH + 10
193
194 isCollide() > if playery > GROUNDY - 50 or ...
```



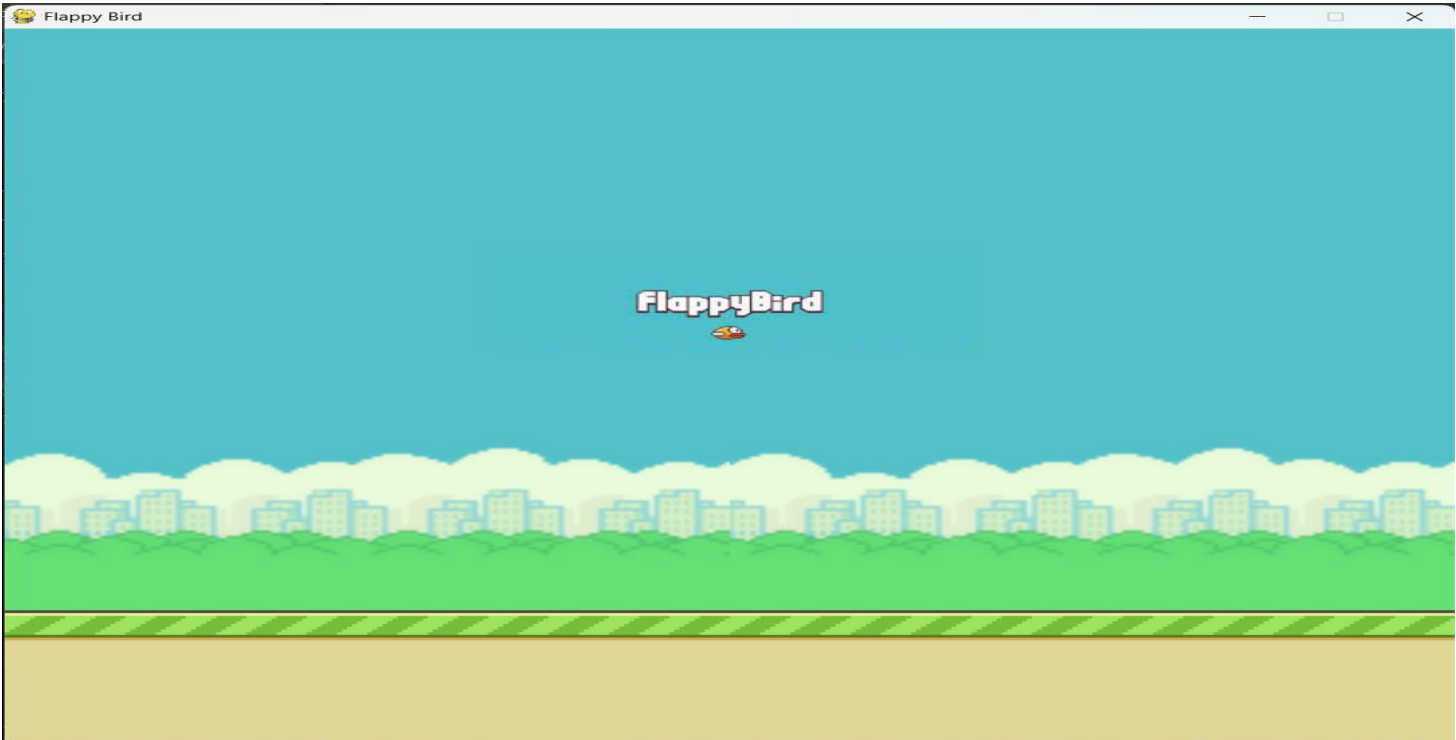
```
142
143     # blitting of pipes
144
145     SCREEN.blit(GAME_SPRITES['background'], (0, 0))
146     for upperPipe, lowerPipe in zip(upperPipes, lowerPipes):
147         SCREEN.blit(GAME_SPRITES['pipe'][0], (upperPipe['x'], upperPipe['y']))
148         SCREEN.blit(GAME_SPRITES['pipe'][1], (lowerPipe['x'], lowerPipe['y']))
149
150     SCREEN.blit(GAME_SPRITES['base'], (basex, GROUNDY))
151     SCREEN.blit(GAME_SPRITES['player'], (playerx, playery))
152     myDigits = [int(x) for x in list(str(score))]
153     width = 0
154     for digit in myDigits:
155         width += GAME_SPRITES['numbers'][digit].get_width()
156     Xoffset = (SCREENWIDTH - width) / 2
157
158     for digit in myDigits:
159         SCREEN.blit(GAME_SPRITES['numbers'][digit], (Xoffset, SCREENHEIGHT * 0.12))
160         Xoffset += GAME_SPRITES['numbers'][digit].get_width()
161     pygame.display.update()
162     FPSCLOCK.tick(FPS)
163
164
165 def isCollide(playerx, playery, upperPipes, lowerPipes):
166     if playery > GROUNDY - 50 or playery < 0:
167         GAME_SOUNDS['hit'].play()
168         pygame.image.load('gallery/sprites/gameover.png').convert_alpha()
169         return True
170
171     for pipe in upperPipes:
```



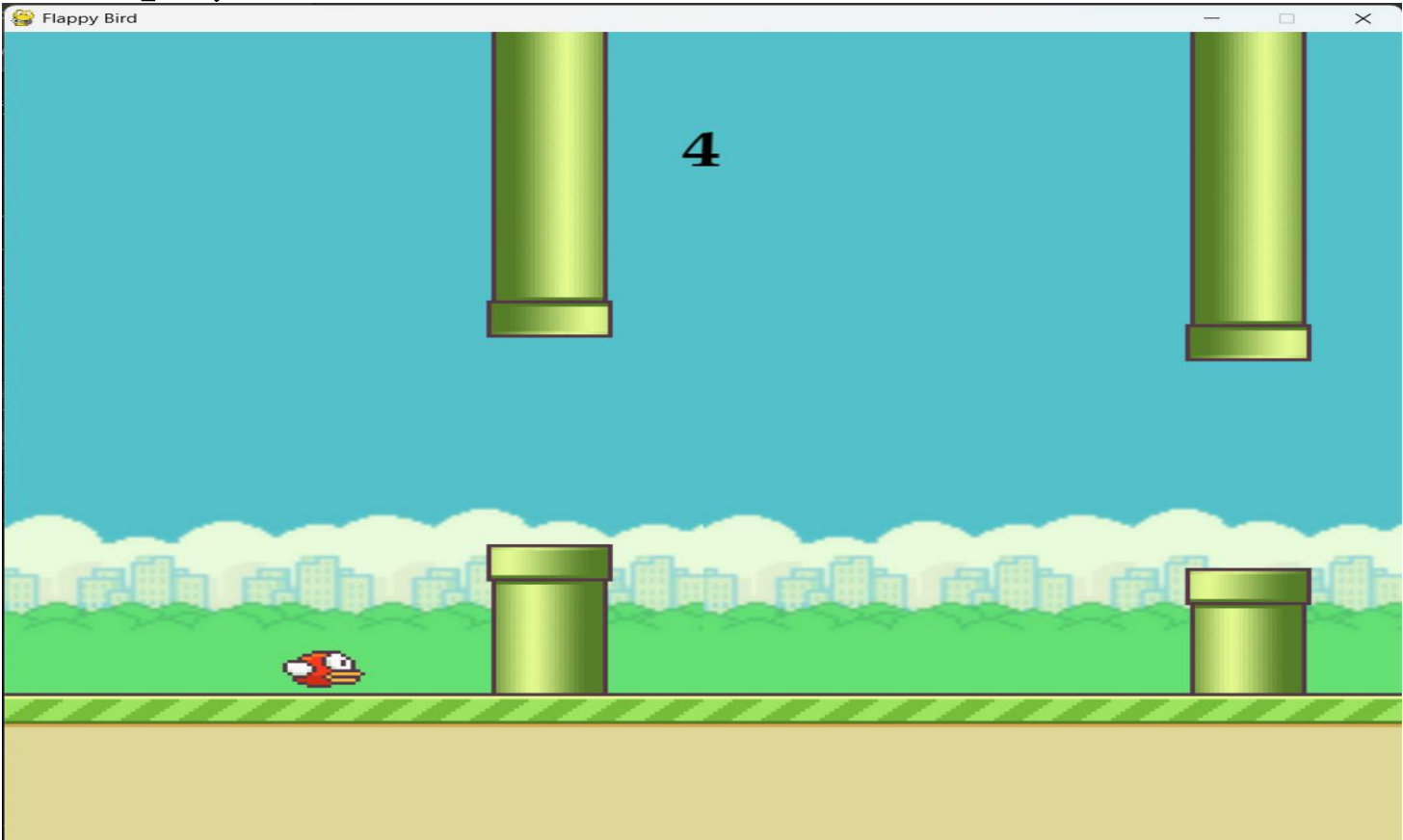
```
115
116     if playerVelY < playerMaxVelY and not playerFlapped:
117         playerVelY += playerAccY
118
119     if playerFlapped:
120         playerFlapped = False
121     playerHeight = GAME_SPRITES['player'].get_height()
122     playery = playery + min(playerVelY, GROUNDY - playery - playerHeight)
123
124     # move pipes to the left
125
126     for upperPipe, lowerPipe in zip(upperPipes, lowerPipes):
127         upperPipe['x'] += pipeVelX
128         lowerPipe['x'] += pipeVelX
129
130     # Add a new pipe when the first is about to cross
131
132     if 0 < upperPipes[0]['x'] < 5:
133         newpipe = getRandomPipe()
134         upperPipes.append(newpipe[0])
135         lowerPipes.append(newpipe[1])
136
137     # removing the pipes if it goes out of the screen
138
139     if upperPipes[0]['x'] < -GAME_SPRITES['pipe'][0].get_width():
140         upperPipes.pop(0)
141         lowerPipes.pop(0)
142
143     # blitting of pipes
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
26
```

GAMEPLAY

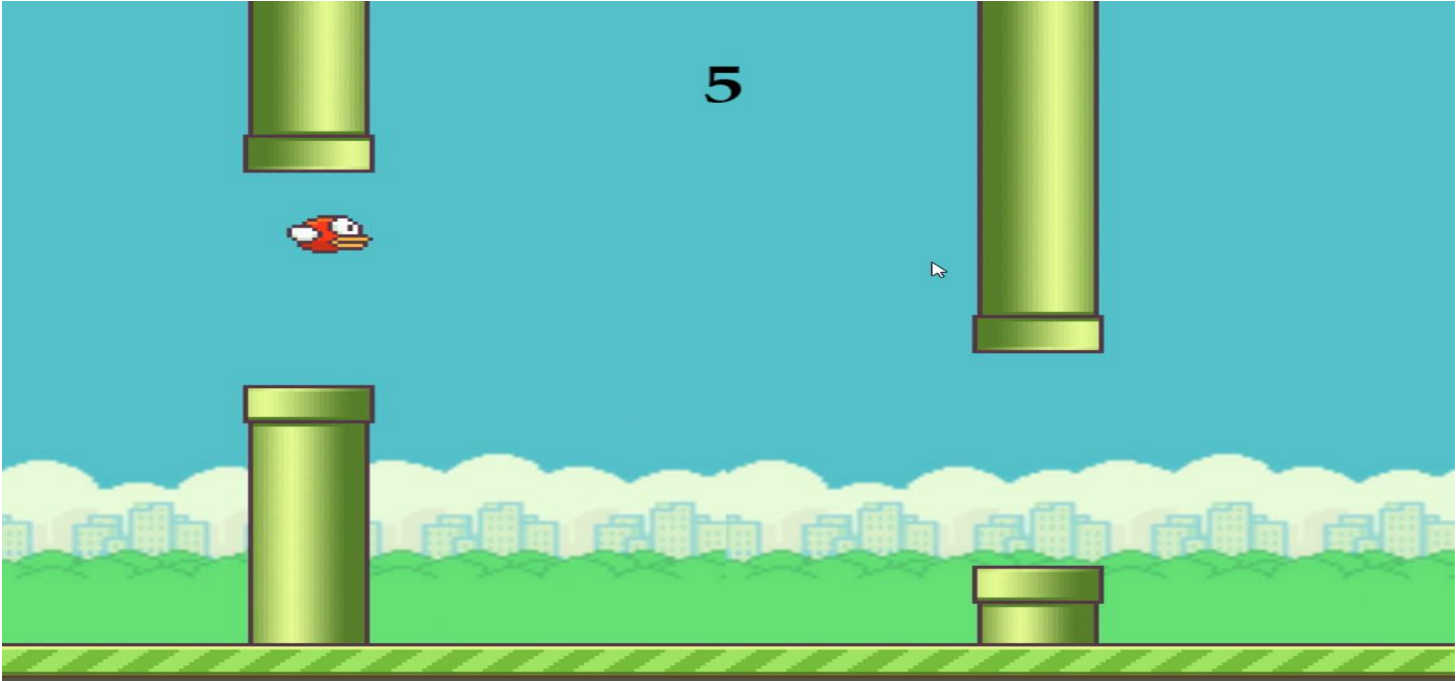
Start Screen and End Screen:-



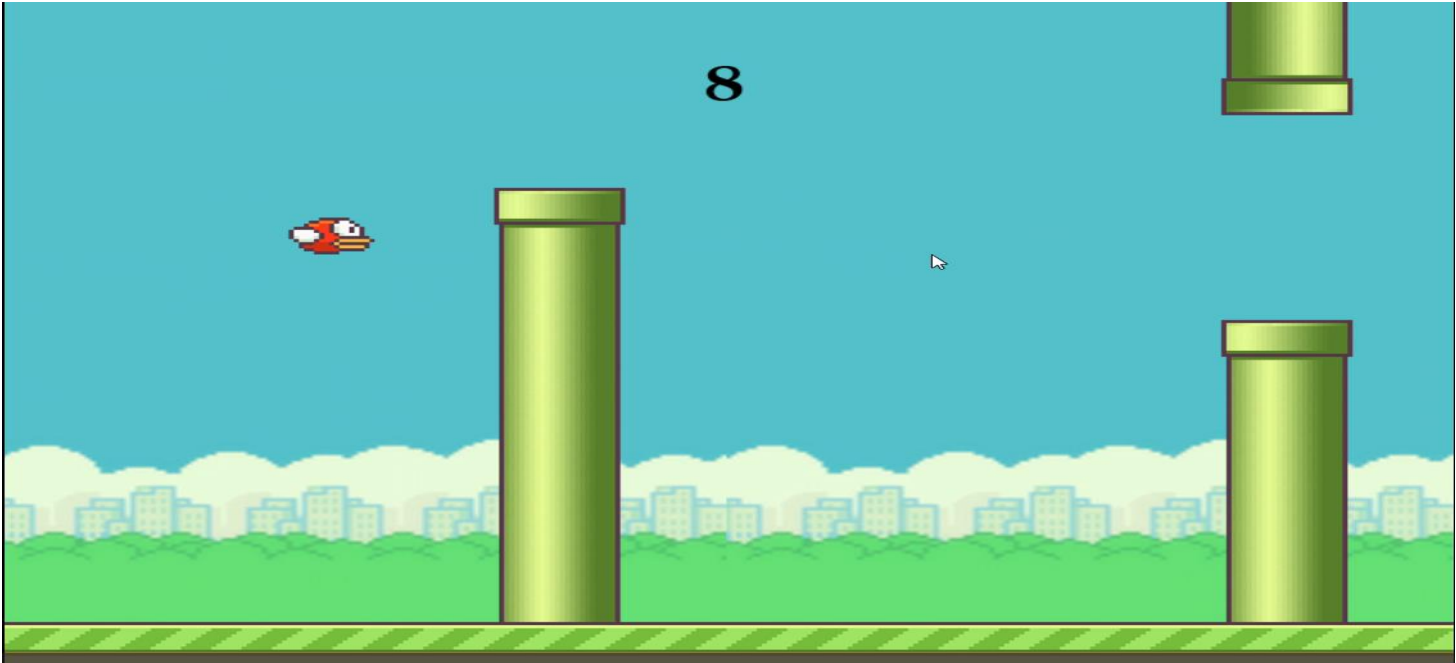
Gameplay-



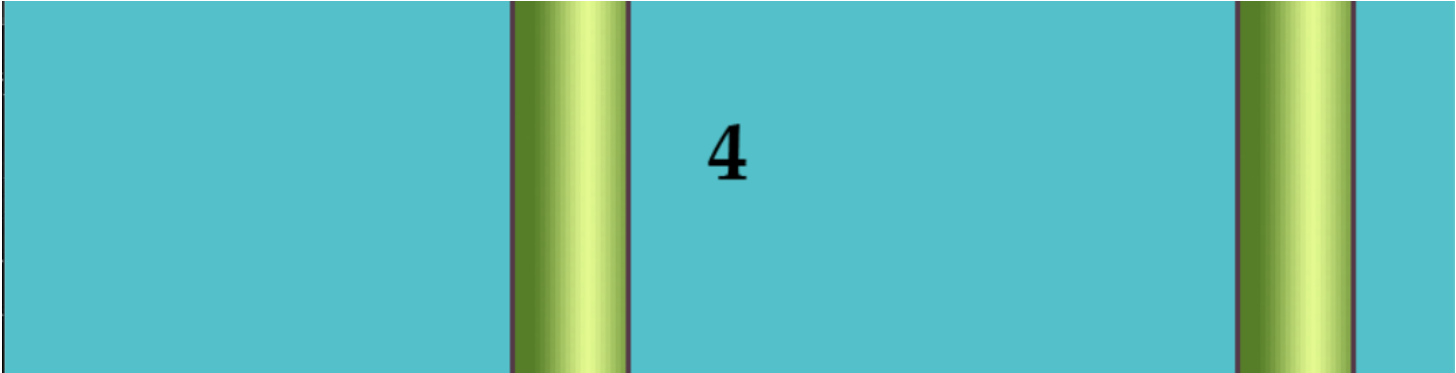
Bird passing through pipe-



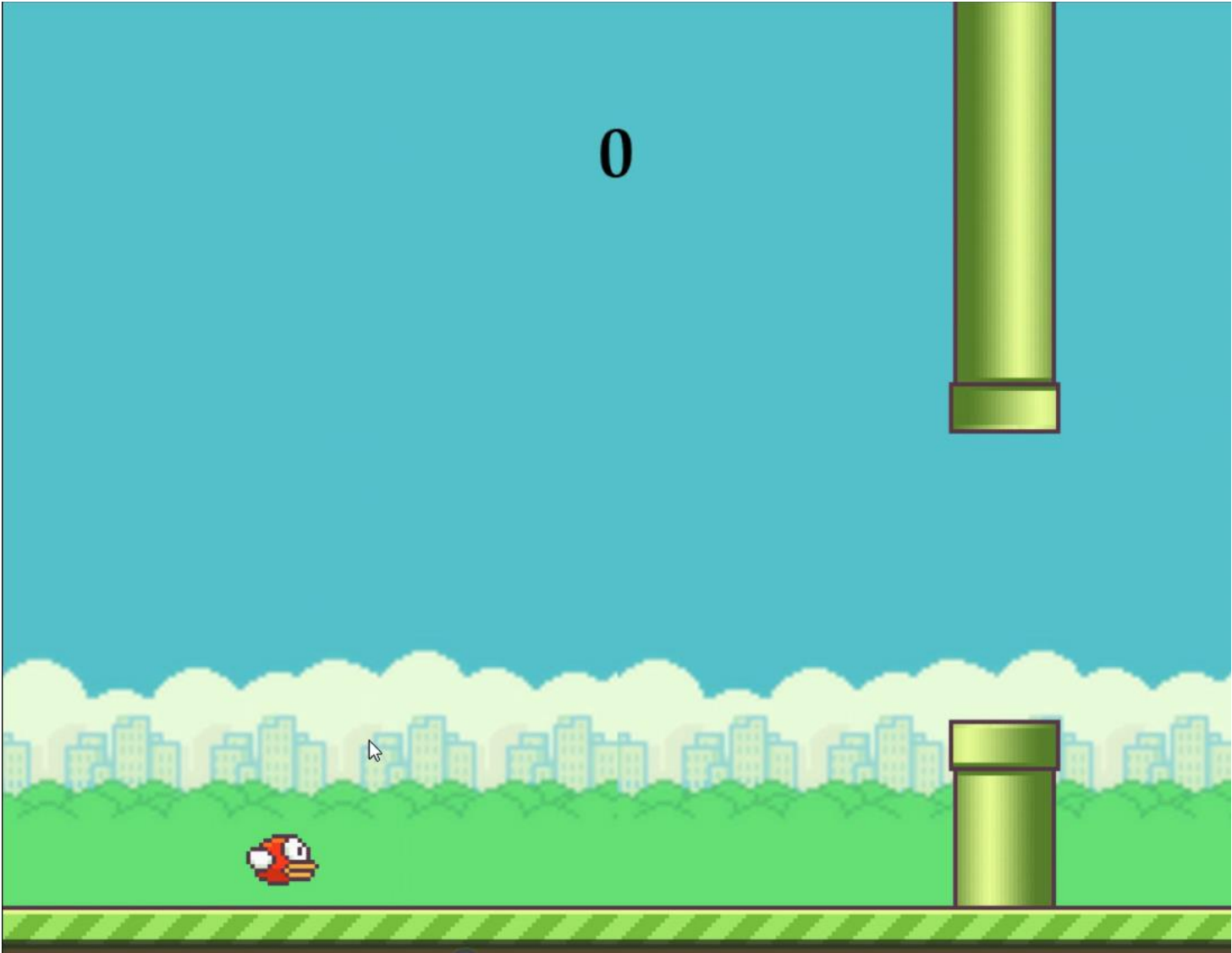
Bird going to collide with pipe-



Score Counter-



Bird going outside the map-



TESTING

Testing refers to the process of evaluating a software application or system to determine whether it meets specified requirements and performs as expected.

Testing is an essential part of software development as it helps to identify defects, errors, or other issues that may affect the software's functionality or performance. The testing process involves running various tests on the software, including functional, performance, security, and usability testing, among others.

The primary goal of testing is to ensure that the software meets the end-users' requirements and is of high quality. It also helps to reduce the risk of defects or errors in the software and improve its overall reliability and maintainability. Testing is typically performed throughout the software development life cycle, from the initial design and development stages to the final release and maintenance stages.

Testing is a crucial aspect of software project management that involves the evaluation of a software product to identify defects, errors, or other issues that might affect its quality, functionality, or performance. The goal of testing is to ensure that the software meets the specified requirements and performs as expected in a variety of scenarios and situations.

There are different types of testing techniques and methods that are used in software project management, including unit testing, integration testing, system testing, acceptance testing, and regression testing. These techniques and methods are employed at different stages of the software development lifecycle to ensure that the software is tested thoroughly and any issues are identified and resolved early on.

Testing plays a critical role in software project management, as it helps to reduce the risk of defects and errors in the final product, improves the quality and reliability of the software, and enhances customer satisfaction. Effective testing practices can help to ensure that the software meets the requirements of the end-users and performs as expected in real-world scenarios.

Our game is fully tested and no kind of errors or problems are found and is ready to use.

CONCLUSION

The Flappy bird game was implemented successfully using a high level scripting language .The game logic was deployed successfully and the gameplay was also smooth .The game is suitable for all age groups and its availability on desktop broadens the scope of its usage .The game is fairly simple and we have tried to add more features to the original classic game. Nevertheless, there is still some scope to add more features like creating multiple logins or making multi-player compatible game but that is beyond the scope of the script that we have been using. We learned a lot about object oriented approach, configuration with other platforms and above all concept of graphics as used in computers. Our course curriculum helped us explore more about the project we are dwelling into and hence give us a wider perspective of how graphics and simple coding gives us a user experience so soothing and something to cherish over and over .

REFERENCES

https://en.wikipedia.org/wiki/Flappy_Bird

<https://flappybird.io/>

<https://www.youtube.com/@CodeWithHarry>

<https://www.youtube.com/@CodingWithRuss>

