A
Project Report
On
"Space Invaders Game"

Prepared by

Rutika Mehta(18DCS045)
Sarthak Prajapati(18DCS095)
Rushil Shah(18DCS111)

Under the guidance of

Prof. Rima Patel

A Report Submitted to

Charotar University of Science and Technology

For Partial Fulfillment of the Requirements for the

5th Semester Summer Internship-I (CS343)

Submitted at

Department of Computer Science & Engineering

Devang Patel Institute of Advance Technology and Research

At: Changa, Dist: Anand – 388421

June 2020

CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CERTIFICATE

This is to certify that the report entitled "**Space Invaders Game**" is a bonafide work carried out by **Rutika Mehta (18DCS045), Sarthak Prajapati (18DCS095) and Rushil Shah (18DCS111)** under the guidance and supervision of **Prof. Rima Patel** for the subject **CS343 Summer Internship-I** of 5thSemester of Bachelor of Technology in **Department of Computer Science & Engineering, DEPSTAR** at Faculty of Technology & Engineering – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred to the examiner.

Prof. Rima Patel
Assistant Professor cum Research Fellow
Department of Computer Science & Engineering
DEPSTAR, CHARUSAT, Changa, Gujarat.

Prof. ParthGoel
I/C Head of Department
Department of Computer Science & Engineering
DEPSTAR, CHARUSAT, Changa, Gujarat

Dr. Amit Ganatra
Principal, DEPSTAR
Dean, FTE
CHARUSAT, Changa, Gujarat.

Devang Patel Institute of Advance Technology And Research,

Changa, Ta.Petlad, Dist.Anand, PIN: 388 421. Gujarat

ACKNOWLEDGEMENT

We, the developers of Space Invaders Game, with immense pleasure and commitment would like to present the project assignment. The development of this project has given us wide opportunity to think, implement and interact with various aspects of management skills as well as the new emerging technologies.

Every work that one completes successfully stands on the constant encouragement, good will and support of the people around. We hereby avail this opportunity to express our gratitude to number of people who extended their valuable time, full support and cooperation in developing the project.

We express deep sense of gratitude towards our Supervisor and Mentor, Ms. Rima Patel, and Head of the Computer Science and Engineering Department, Mr. Parth Goel for the support during the whole session of study and development. It is because of them, that we were prompted to do hard work, adopting new technologies.

They altogether provided us favorable environment, and without them it would not have been possible to achieve our goal.

Thanks,
Rutika Mehta
Sarthak Prajapati
Rushil Shah

ABSTRACT

The aim of this project is to build a 2D arcade like game from scratch in Python. The core of the game is PyGame, a set of Python modules designed for game development, which is used for tasks such blitting images on the screen or moving the said images. The end product is a five levels game, the goal of the player being to destroy all the enemy space ships with bullets, encountering intelligent enemies in the latter level of the game.

The following report will provide an in depth understanding of the technicality of the project. The report describes the context of this work and steps of implementation. It also describes the approaches undertaken during the development. The report concludes with a reflection of the last few months of work and the outcome of the project.

To sum it up, the report is to be used as a guide on a journey that will answer the following questions about Space Invaders Game: ‘Why and how the project was build?’ and ‘What interesting features the game has and why?’

TABLE OF CONTENTS

CHAPTERS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 - PROJECT DEFINITION

The game, called Space Invaders, is a 2D arcade like space shooter built with Python and elements of PyGame. The main idea of the game is for the player to destroy all the enemy spaceships by shooting bullets and reach the last level of the game where intelligent enemy alien will do the absolute best to defeat the player spaceship. Along the intermediate levels, the alien spaceships will try to knockout the player by either colliding with it or shooting bullets.

The player spaceship is controlled via keyboard and the enemy spaceships have a predefined movement. The player has to dodge enemy bullets in order to survive the continuous attack coming from the opponent forces.

CHAPTER 2 - DESCRIPTION

The game starts with the home screen that displays “Space Invaders”, which is the name of the game, along with a “Play” option. Clicking on the “Play” option will take the player to level 1.

Fig. 2.1 Home Screen

The screen will display the level number for 3 seconds and then the level will start. Level 1 is easy because of slow speed and a smaller number of enemies.

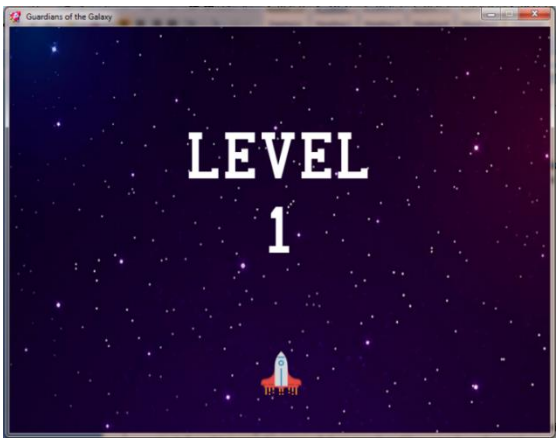


Fig. 2.2 Level 1

Fig. 2.3 Level 1 Gameplay

The player can move left or right using the left and right arrow keys on the keyboard, and shoot the bullets using spacebar. The enemies will move in horizontal direction and when it reaches the boundary, it comes one step closer to the player spaceship’s axis. The player has to shoot the bullet and if it hits an enemy, then that enemy would disappear and another enemy would respawn at a random place (far from the player spaceship’s axis).
Time limit is also set which is displayed on the upper right corner of the screen. If the player does not match a particular score within the time limit, the game is over.

Reaching a score of 10 will take the player to level 2. Here, the speed and the number of enemy is increased. The look of the enemy is also changed. Along with difficulty, the time limit is also increased.

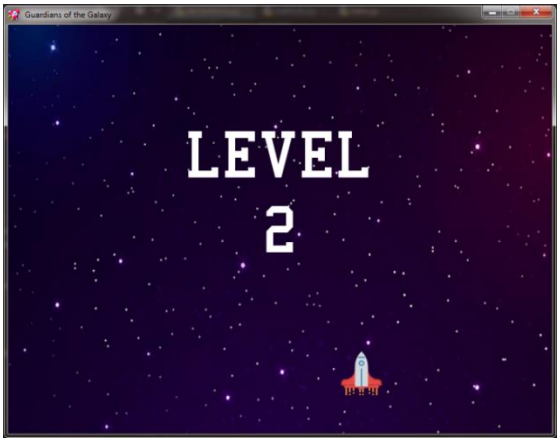


Fig. 2.4 Level 2

Fig. 2.5 Level 2 Gameplay

Reaching a score of 20 will take the player to level 3. Here, the speed and the number of enemies is limited but the enemy can shoot bullet on the player spaceship. If hit by the enemy bullet, the game is over. The time limit is also increased.

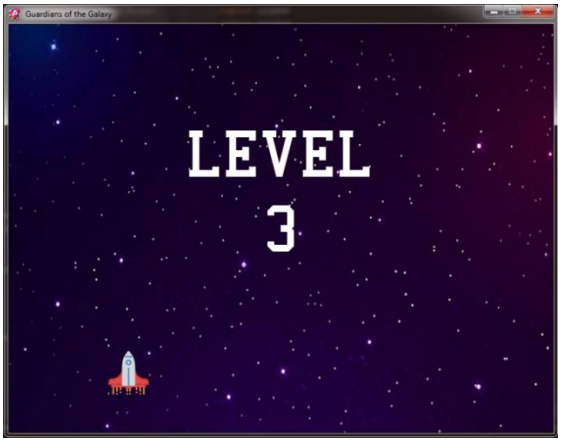


Fig. 2.6 Level 3

Fig. 2.7 Level 3 Gameplay

Reaching a score of 30 will take the player to level 4. Here, the speed and the number of enemy is increased as well as enemy can shoot bullet on the player spaceship. If hit by the enemy bullet, the game is over. The time limit is also increased.

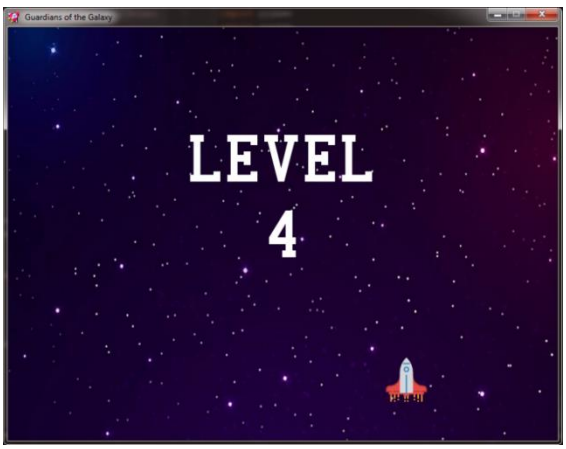


Fig. 2.8 Level 4

Fig. 2.9 Level 4 Gameplay

A score of 40 will take the player to the final level of the game. This level contains single powerful enemy which changes its positions very frequently and shoots the bullets at very high speed. The player wins the game if the final level enemy is hit 10 times. Time limit is also increased.

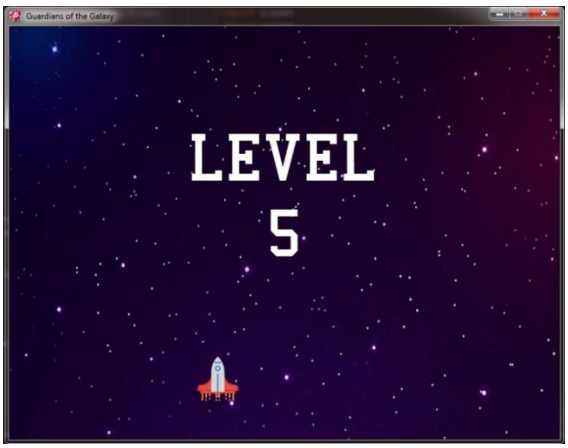


Fig. 2.10 Level 5

Fig. 2.11 Level 5 Gameplay

Score of 50 means the player has won the game. The screen says that the player has won the game and clicking on “Play Again” option will start the game from beginning with all score set to zero. Game over will display the screen shown below on the right side.

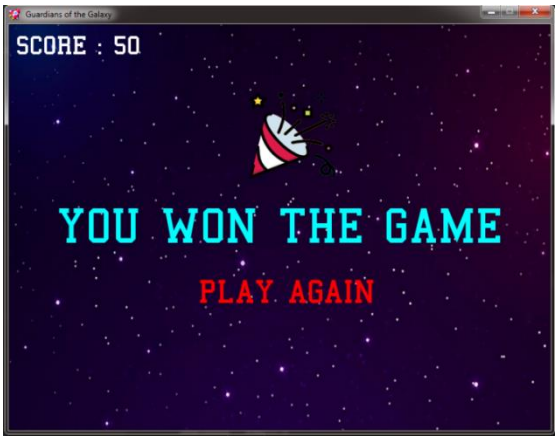


Fig. 2.12 Won the Game

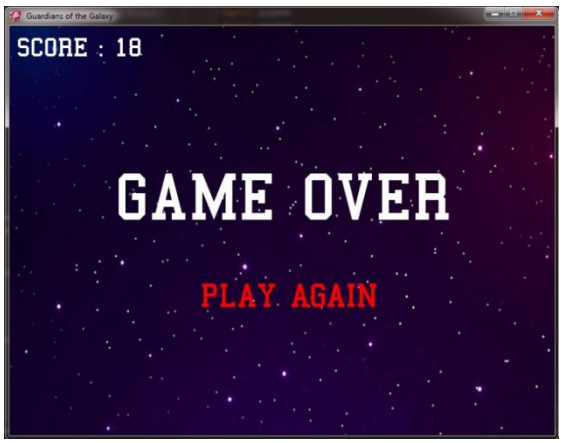


Fig. 2.13 Game Over

CHAPTER 3 - SOFTWARE AND HARDWARE REQUIREMENTS

- PyCharm / Jupyter / Visual Studio Code (IDE)
- PyGame library
- Python Interpreter Setup
- Operating System: Windows 7 or higher / Mac
- Processor: Core i3 or higher
- RAM: 2 GB or higher
- Size on disk: 39.7 MB

CHAPTER 4 - MAJOR FUNCTIONALITY

4.1 COLLISION DETECTION

The collision detection between the enemy and the player spaceship, or the collision detection between bullets and the player/enemy is the crucial part of this game.
This is implemented using the mathematical formula to find distance between two given coordinates.

Let the given coordinates be (x1,y1) and (x2,y2)
Let distance between these coordinates be d
Thus,

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (4.1.1)$$

The position of enemy, bullet, or spaceship at any given instance can be represented in terms of (x,y) coordinates. Thus, distance between enemy, bullet or spaceship can be calculated using above formula.
For levels 1 to 4, if the distance is less than 27 pixels, then collision has occurred. For level 5, if the distance is less than 45 pixels, then collision has occurred.

```
# Function to check the collision of bullet with enemy or the collision of enemy bullet with the player
def isCollision(x1,y1,x2,y2):
    distance = math.sqrt((math.pow(x1 - x2,2)) + (math.pow(y1 - y2,2)))
    if distance < 27:
        return True
    else:
        return False

# Function to check the collision of bullet with enemy or the collision of enemy bullet with the player in level 5
def isCollision5(x1,y1,x2,y2):
    distance = math.sqrt((math.pow(x1 - x2,2)) + (math.pow(y1 - y2,2)))
    if distance < 45:
        return True
    else:
        return False
```

Fig. 4.1 Collision Detection Implementation

4.2 TIME LIMIT

For the time limit functionality, the time is captured at the beginning of each level. It is then compared to the predefined time limit set for each level. The time left for the player is also decreased at the rate of one unit per sec.

```
# Check if timer's up or not
if start_time + (play_time1 * 1000) <= pygame.time.get_ticks():
    game_over()
    mixer.Sound('gameover_low_level.wav').play()
    flag = 1
    for j in range(num_of_enemies1):
        enemy1Y[j] = 1000
    game_ended = True

if not game_ended:
    # Calculating time left
    timeLeft = pygame.time.get_ticks() - start_time
    timeLeft = timeLeft / 1000
    timeLeft = play_time1 - timeLeft
    timeLeft = int(timeLeft)
    # Function to display time remaining
    time_left(timeLeft)
```

Fig. 4.2 Time Limit Implementation

4.3 MULTIPLE LEVELS

The difficulty and the layout of the game changes according to the levels. The levels change based on the score. If the player bullet hits the enemy (use of collision detection), the score is increased by one unit.

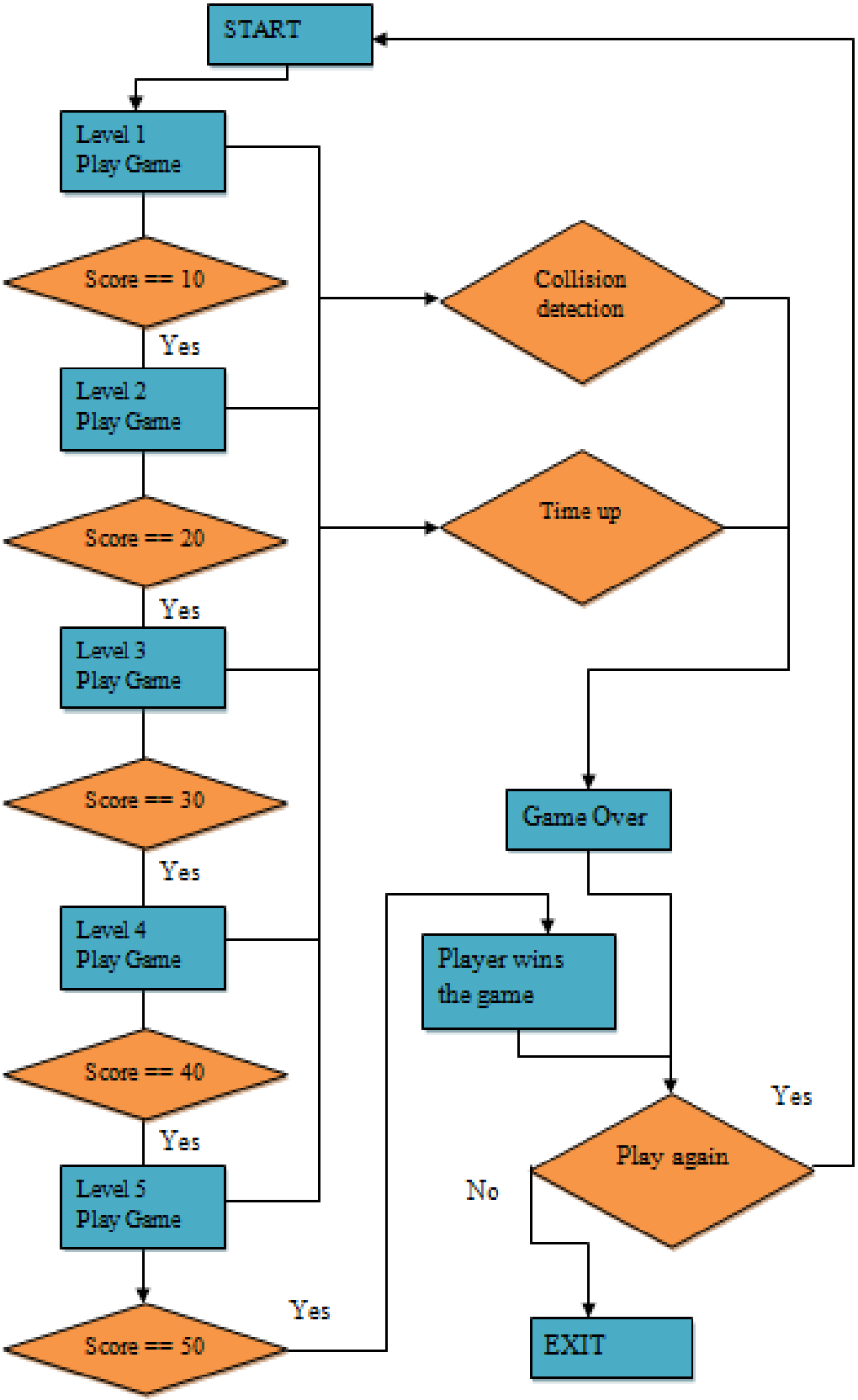
Table 4.1 Level number according to the Score

Score range	Level
0 - 9	1
10 - 19	2
20 - 29	3
30 - 39	4
40 - 49	5

```
# Changing level depending upon the score
display_score(10,10)
if score == 10 and level == 1:
    level = 2
if score == 20 and level == 2:
    level = 3
if score == 30 and level == 3:
    level = 4
if score == 40 and level == 4:
    level = 5
if score == 50 and level == 5:
    flag = 2
```

Fig. 4.3 Multiple Levels Implementation

CHAPTER 5 - FLOW CHART



CHAPTER 6 - SCREENSHOTS OF THE PROJECT OUTPUT

Fig. 6.1 Home Screen

Fig. 6.2 Level 1

Fig. 6.3 Level 1 Gameplay

Fig. 6.4 Level 2 Gameplay

Fig. 6.5 Level 5 Gameplay

Fig. 6.6 Level 3 Gameplay

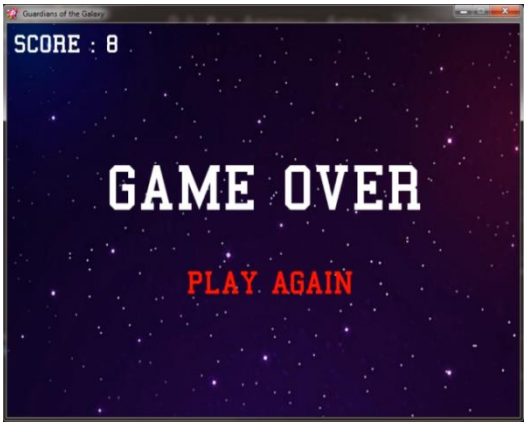


Fig. 6.7 Game Over

Fig. 6.8 Won the Game

CHAPTER 7 - LIMITATIONS OF THE PROJECT

- Player cannot pause the game
- Player cannot mute the background music
- The game contains only 5 levels
- Requirement of Python setup, Python Editor, PyGame library to execute the code
- It can be played on desktop PCs or laptops only
- Single player game
- Player spaceship has only one life (single chance) throughout all the levels
- If game is over at a particular level, then the player has to start again from level 1
- The game does not contain High Score option
- The speed of the player spaceship remains constant throughout all the levels

CHAPTER 8 - OUTCOME (CONCLUSION)

The outcome of this project was a 2D arcade like Space Invaders game and a lot of new things to learn. Looking back to the past few months, it was a challenging task initially, without much progress due to the lack of technical knowledge required for this project, but took off while making progress with the research. We began the project with absolutely no idea on how graphics work, how a collision detection system works or even how to display an image on the screen and, by the end of the project, we got to a point where we could create our own or modify existing game specific algorithms to suit our needs.

CHAPTER 9 - FUTURE ENHANCEMENTS

- Making an exe file of the game
- Adding pause button
- Adding an option to mute the background music
- Adding power ups based on the scores
- Adding life (health bar) so that the player can survive more than one attack
- Adding functionality such that the state of the player where he/she reached the last time is saved. Thus, the player does not have to start all over again from level 1
- Adding High Score option
- Changing the speed of the player spaceship as per the levels
- Making this a multiplayer game
- Making the game online (Cloud based)
- Adding more levels and challenges

CHAPTER 10 - REFERENCES

Web reference:

[1] https://www.pygame.org/docs/

[2] https://www.dafont.com/

[3] https://www.flaticon.com/

[4] https://en.wikipedia.org/wiki/Space_Invaders

[5] https://www.youtube.com/watch?v=FfWpgLFMI7w