PACMAN

A Project Work

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING IN

CSE-AIML

NAME OF THE STUDENT

Aashish Kalra

University Roll Number

20BCS6852

Under the Supervision of:

Mr. Digvijay Puri

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING APEX INSTITUE OF TECHNOLOGY

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI, 140413,

PUNJAB

29/07/2021

ACKNOWLEDGEMENT

I extend our sincere and heartfelt thanks to our esteemed guide, **Mr. Digvijay Puri** and for his exemplary guidance, monitoring and constant encouragement throughout the course at crucial junctures and for showing us the right way.

I am grateful to respected coordinator **Er. Akwinder Kaur** for permitting me to utilize all the necessary facilities of the Institute.

We would like to thank other faculty members also.

Last but not the least,

I would like to thank my friends and family for the support and encouragement they have given us during the course of our work.

Aashish Kalra

20BCS6852

ABSTRACT

This project discusses about the popular Pacman game using C and C++. We built a simple Pacman implementation with the maze, a Pac-man and Pac-dots for the Pac-man to eat. The Pac-man moves around legal positions randomly and eats the Pac-dots. We added a ghost. With checking to see if the game has ended (ghost catches Pacman) this second increment was completed. We also added a verification process in which user need to enter the name and then you can start to play the game and we also added the point feature in which user will get to know his/her points.

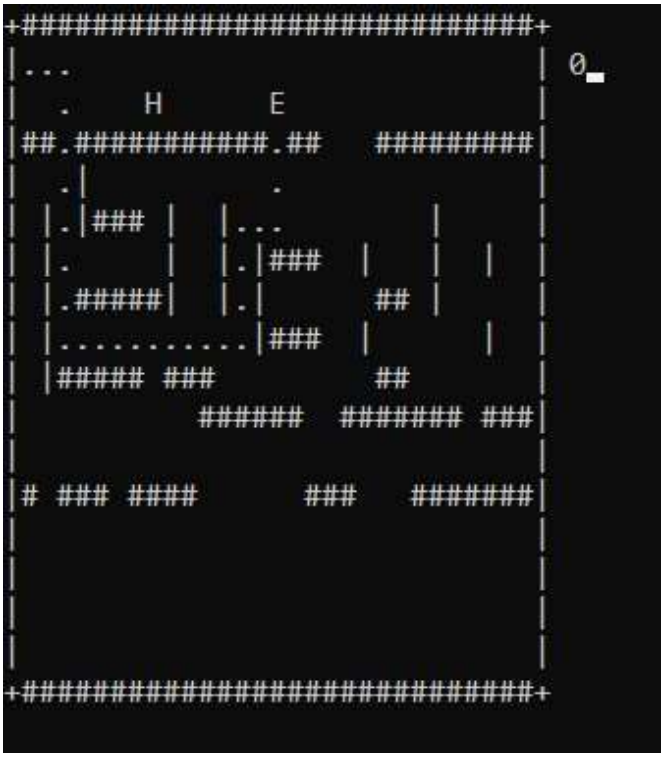
INDEX

LIST OF FIGURES

1. INTRODUCTION

Pac-man is a game in which Pac-man is an agent that eats the Pac-dots. Ghost is an agent that attack the Pac-man. The whole game is played in maze environment.

In this project, I've designed agents for the classic version of Pac-man. Along the way, I've also implemented minimax algorithm.



E= Eater (Pac-man)

H= Handler (Agent)

Fig 1 : Pac-man environment

2. LITERATURE SURVEY

Past System:

'Pac-Man' used to be played on Arcade-gaming machine, it became famous at that time and then it was introduced in a PSP and people used to enjoy Pac-Man in that as well.

Existing System:

The game 'Pac-man' was originally called and launched as 'Puck man', according to its Japanese title. After several changes, the game exists as a maze chase game with four coloured ghosts namely, Blinky (red), Pinky (pink), Inky (cyan) and Clyde (orange). The objective of the game is to eat all of the dots placed in the maze and the player advances to another level when Pac-man eats all the dots.

Future System:

'Pac-Man' might be get introduced in a 3D version and even it can also be played by using AR(Augmented Reality) as people are already working to make these thing possible and it will become more and more interesting when this game will keep on eveolving.

3. DESIGN

3.1 Software Requirements Specification

This section describes the intended purpose, requirements and nature of a software developed. Software requirements specification (SRS) is a description of a software system to be developed, its defined after business requirements specification. The SRS lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

3.1.1 Purpose

The purpose of this document is to build an application which is used to provide stationary and printed materials easily.

3.1.2 Requirements to Develop / Rest Web Services Application on PC

Hardware Requirements

• Microsoft Windows 7/8/10 (32-bit or 64-bit)

• 2 GB RAM minimum.

• 1 GB of available disk space minimum.

• 1280 x 800 minimum screen resolution

Software Requirements

• Dev-C++ or any other IDE.

• C and C++ (programming language)

4. PROBLEM FORMULATION

- Pacman has been awarded eight records in *Guinness World Records: Gamer's Edition 2008;* given the title *"Most Successful Coin-Operated Game".* The game dates back to 1980's and has a legacy till date.
- It is one of the most influential video games of all times. Established in the maze chase game genre, it applies AI that reacts to player actions as ghosts(eater) chases pac-man(handler).
- Pac-Man was one of the earliest successful non-shooting action games, introducing concepts like "parallel visual processing," which entails keeping track of many entities at the same time, such as the player's location, attackers, and energizers.
- Pac-Man also inspired 3D games like Monster Maze (1982) and Spectre (1982), as well as early first-person shooters like MIDI Maze.

5. OBJECTIVES

- To surpass the bounds of arcade game genre and appeal the female audience and other age groups as it is a maze chase game.
- To provide gamers with an experience of AI applied in gaming and open space for creativity by setting a roadmap in this direction.
- To act as a stress buster for children and even adults of various age groups.
- To enhance concepts like parallel visual processing.

6. METHODOLOGY

- We have defined several methods to perform different functions and we have directed the flow to them as required.
- The program starts with a simple command to enter the name of the user in order to validate.
- Then we have fixed the coordinates of *the handler* and *the eater* which are saved in the respective variables.
- Next, we print the instructions on the terminal and ask the user to enter the difficulty level of the game.
- We have made use of *if-else-if* loop to adjust the speed of the eater.
- Next, we clear the screen and there's a call to function *ShowMap* which prints the maze on the terminal.
- We, then, make a call to *gotoxy* function which sets the console control of the maze, followed by printing of the *handler(H).*
- *FindPath* method is called next which basically helps the eater to find the shortest path to the handler's current position.
- We have also made use of *while* loop. Under this, we've set the instructions for the program to eat pac-dots by using *if-else-if* loop to identify the command received from the user.
- As the handler moves, there's a creation of space on its old co-ordinates and printing of H on its new position.
- Next, we are creating the trail of pac-dots as eater moves.
- Since, the game only ends when the position of handler and eater is same, we have made use of an *if* statement to judge the operation.

- Since, we are also keeping the score, we have fixed the place of scoreboard which keeps the track of the number of pac-dots handler eats.
- Again, we have made use of another *if-else* statement to judge whether the score is higher than 999.
- Lastly, we print the result.
- It's better illustrated in the UML diagram.

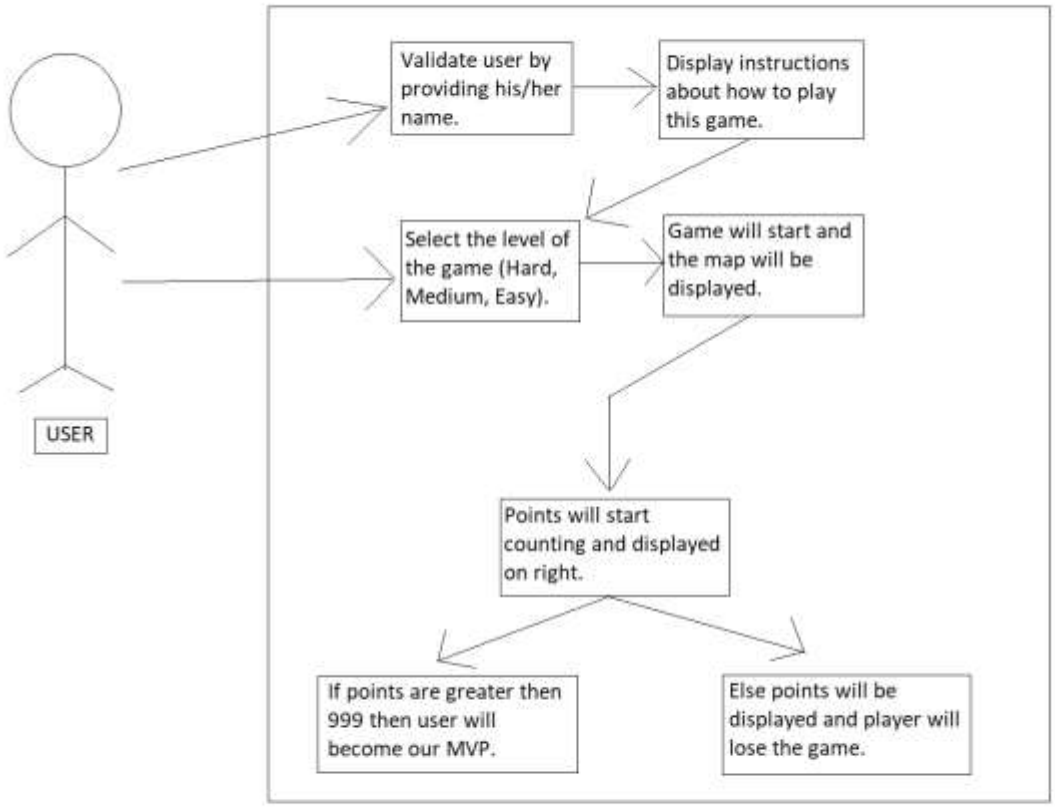


Figure 2: UML Diagram (Use-Case Diagram)

7. RESULTS

Figure 3 : Verification Menu

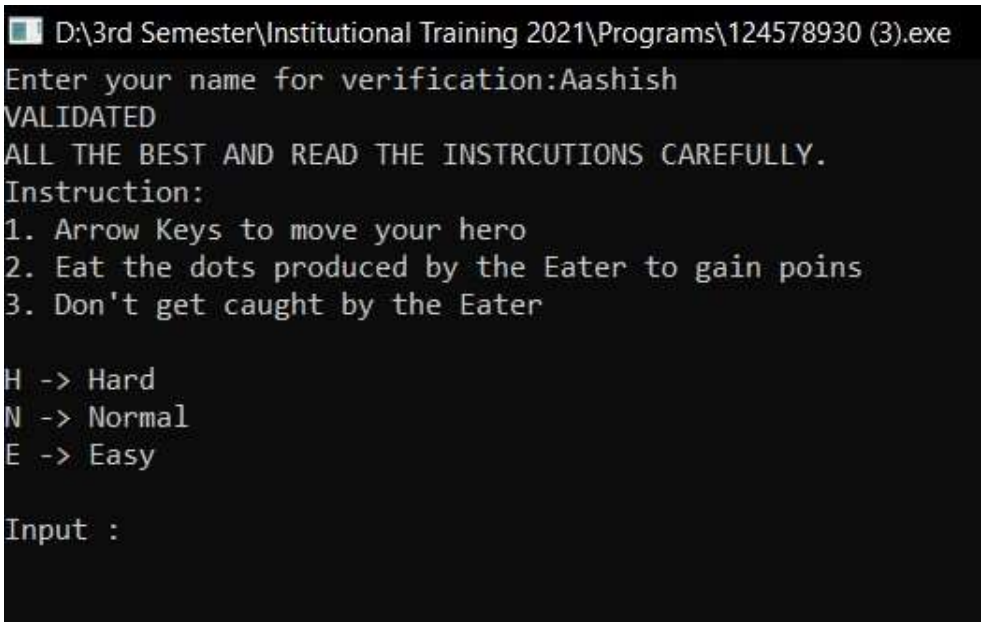


Figure 4 : Main Menu

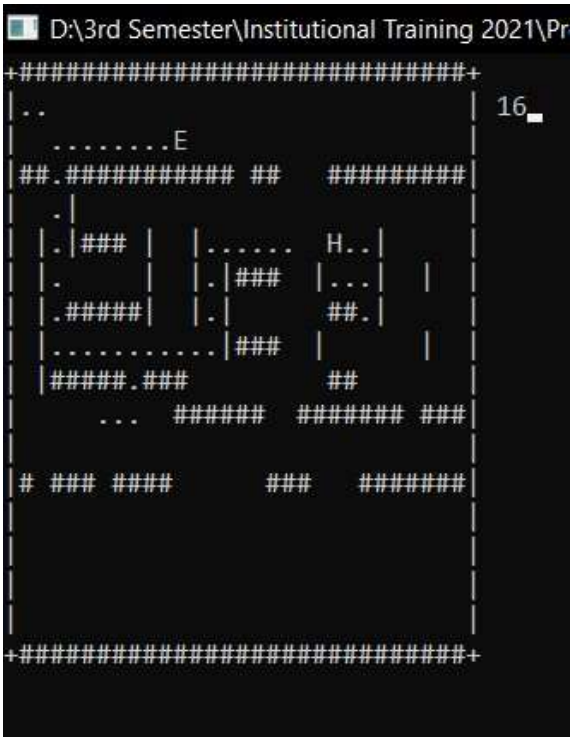


Figure 5 : Game Processing

If points is greater than 999:

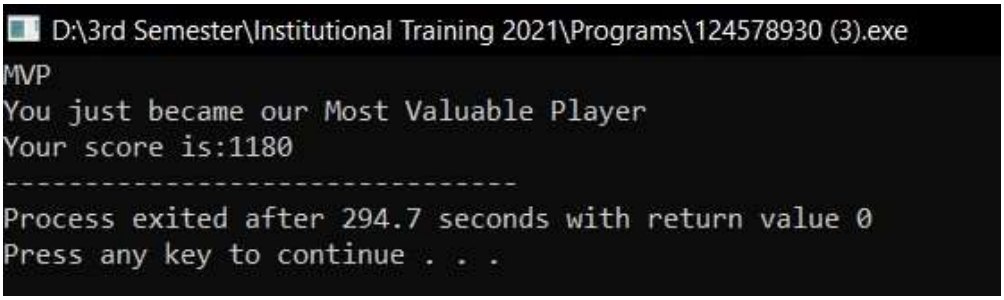


Figure 6: Result (If score is greater then 999).

If points is less than 999:

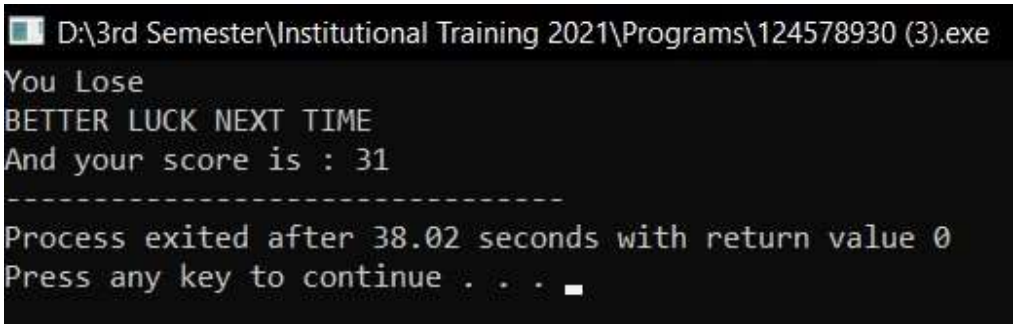


Figure 7: Result (If score is less then 999).

8. CONCLUSION

The player who succeeds in eating more dots by avoiding ghosts scores much points. The code I wrote is for the IDE. This can be extended and can be used in mobile applications so that it will be very flexible for the user to play the game.

The project which I undertaken has helped me gain a better perspective on various aspects related to my course of study as well as particular knowledge of particular web-based applications. I became familiar with software analysis, designing Sprite Sheets, implementation, testing and maintenance considered with my project.

9. REFERENCES

- www.stackoverflow.com
- www.tutorialpoint.com
- www.wikipedia.com
- www.slideshare.com
- www.w3schools.com
- www.github.com