MINI PROJECT

(2022-23)

“Snakey”

Mid Term Report

Institute of Engineering & Technology

Submitted By -

Puroo Kulshrestha (201500535)

Under the Supervision of

Mr. Manoj Varshney

Technical Trainer

Department of Computer Engineering & Applications

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Declaration

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"Snakey"**, in partial fulfillment of the requirements forthe award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under thesupervision of **Mr. Manoj Varshney, Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign: *purookulshrestha*

Name of Candidate: Puroo Kulshrestha

University Roll No.: 201500535

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Certificate

This is to certify that the project entitled “Snakey”, carried out in Mini Project – I Lab, is a bonafide work by Puroo Kulshrestha and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (ComputerScience & Engineering).

Signature of Supervisor:

Name of Supervisor: Mr. Manoj Varshney

Date: 25-11-2022

Training Certificates

- **Puroo Kulshrestha**

Department of Computer Engineering and Applications GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,Chaumuha, Mathura – 281406 U.P (India)

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mr. Manoj Varshney, our technical trainer and supervisor.

She has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents and Seniors for helping us to grab this opportunity to get trained.

Thanking You

Sign: *purookulshrestha*

Name of Candidate: Puroo Kulshrestha

University Roll No.:201500535

ABSTRACT

Snake game is a classic video game that has been enjoyed by people of all ages since its inception in the late 1970s. It is a simple game that involves controlling a snake that moves around the screen and eats food. As the snake eats food, it grows longer, and the player's objective is to prevent the snake from colliding with the walls or its own body. In this abstract, we will discuss the development of a snake game using Python programming language.

The snake game was developed using the Pygame library in Python. Pygame is a set of Python modules designed for writing video games. It provides functionality for handling graphics, sound, input, and other game-related tasks. The game was developed by creating a Pygame window and adding a snake and food sprite. The snake was made up of a series of rectangles, and the food was a single rectangle. The game starts by initializing the Pygame library and setting up the window. The snake and food sprites are then created and placed on the screen. The game loop is then started, which is responsible for updating the game state and drawing the screen. The game loop works by continually updating the position of the snake and checking for collisions with the walls and food. If the snake collides with the food, the snake's length is increased, and a new food sprite is randomly placed on the screen. If the snake collides with the wall or its own body, the game ends. The game also includes a score counter that increases as the snake eats food. The score is displayed on the screen along with the game over message when the game ends.

The snake game in Python was developed using object-oriented programming (OOP) principles. The game was broken down into several classes, including the Game class, Snake class, and Food class. Each class had its own set of methods and attributes that were used to implement the game's functionality. The Game class was responsible for managing the game loop and updating the game state. It also handled user input and displayed the score and game over message. The Snake class represented the snake sprite and was responsible for updating the snake's position and checking for collisions. It also handled the snake's movement and growth. The Food class represented the food sprite and was responsible for randomly placing new food on the screen.

Overall, the snake game in Python was a fun and challenging project that helped to reinforce fundamental programming concepts such as OOP, game logic, and Pygame functionality. The completed game was a great example of how Python can be used to develop simple games and applications. With some modifications, the game could be expanded to include additional features such as different game modes, power-ups, and obstacles.

CONTENTS

1. INTRODUCTION

1.1 Context

Python is an interpreted, high-level, general-purpose programming language that was first released in 1991. It is known for its simple syntax and readability, which makes it easy to learn for beginners and popular among developers for building web applications, scientific computing, data analysis, artificial intelligence, and machine learning. However, "snake" can also refer to a classic video game, often called "Snake" or "Nokia Snake," which involves controlling a growing line or snake on a screen and trying to avoid obstacles while collecting food. This game has been popular since the 1970s and has been implemented in many programming languages, including Python. Implementing the Snake game in Python can be a fun and educational exercise for beginners learning the language.

1.2 Client Identification/Need Identification/Identification of relevant Contemporary issue

The client's need is to build a snake game in Python that is interactive and entertaining for the users. The game should have features such as a graphical user interface, a scoring system, the ability to increase the snake's length, and collision detection with walls or obstacles. While the snake game in Python is not directly related to any contemporary issue, it can be used as an educational tool to promote coding skills and programming literacy among individuals, including students and professionals. Additionally, building games and applications using Python is a growing trend in the tech industry, and game development using Python is a popular choice among developers. Therefore, building a snake game in Python can help individuals develop relevant technical skills that can be useful in various industries.

1.3 Identification of Problem

Identifying a problem related to a snake game in Python can be challenging, as it is primarily a programming exercise or a fun project for beginners to learn Python

programming language. However, some potential problems that developers may encounter when building a snake game in Python include:

1. Performance issues: As the snake grows in length and the game progresses, the speed and performance of the game can decrease. This can result in a slower and less responsive game experience for the user.

2. User Interface Design: Designing a visually appealing and user-friendly graphical interface for the game can be challenging, especially for beginner developers.

3. Collision Detection: Detecting collisions between the snake and obstacles or walls can be a challenging task, and improperly implemented collision detection can result in an unfair or inconsistent gameplay experience for the user.

4. Difficulty Balancing: Balancing the difficulty of the game can be a tricky task, as the game should be challenging enough to keep users engaged but not so difficult that it becomes frustrating.

5. Code Complexity: Depending on the developer's level of expertise, implementing a snake game in Python can be a complex task, and writing clean, efficient, and scalable code can be a challenge.

1.4 Identification of Tasks

Identifying the tasks involved in building a snake game in Python will depend on the complexity of the game and the developer's approach. However, here are some potential tasks that developers may need to undertake when building a snake game in Python. The developer needs to install and set up the Python development environment and necessary libraries or frameworks such as Pygame. The developer needs to design the graphical user interface, including the game board, snake, food, and other game elements. The developer needs to implement the game logic, including handling user input, moving the snake, detecting collisions with obstacles and walls, generating food, and updating the score. The developer needs to test the game thoroughly, identify and fix any bugs or issues that may arise during testing. The developer should document the game, including the requirements, design, implementation, and testing procedures.Once the game is functional, the developer can refine and polish the game by improving the user interface, adding more features, and balancing the difficulty level.

1.5 Timeline

Week 1:

- Research and familiarize yourself with Python language and programming concepts.
- Install necessary tools and frameworks such as Python, Pygame, and a text editor or an IDE.
- Design the game board and basic game interface using Pygame.

Week 2-3:

- Implement the basic logic of the game such as moving the snake, generating food, and detecting collisions.
- Add a scoring system to the game, which keeps track of the score as the snake eats food.
- Test and debug the game for issues and bugs.

Week 4:

- Document the game, including the requirements, design, implementation, and testing procedures.

1.6 SOURCES

The source of our project (including all the project work, documentations and presentations) will is available at the following link **https://github.com/purookulsh13/Snake_Game-**.

CHAPTER -2

LITERATURE REVIEW/BACKGROUND STUDY

2.1. Timeline of the reported problem

When conducting a literature review or background study on the reported problem of a snake game in Python related to game performance and responsiveness, the timeline can vary depending on the available resources and the scope of the problem. Here is a possible timeline for conducting a literature review or background study:

Week 1:

Identify relevant academic and industry publications related to game development, Python programming, and performance optimization. Read and analyze the publications to gain a better understanding of the common performance issues in game development, best practices for Python programming, and techniques for performance optimization.

Week 2:

Search online forums, communities, and blogs related to Python game development to gain insights and practical tips from developers who have encountered similar issues. Identify open-source projects or libraries that can be used to optimize Python game development.

Week 3:

Conduct interviews with experienced Python game developers to gain insights into their approaches to optimizing game performance. Investigate performance profiling tools and techniques to identify performance bottlenecks in the game code.

Week 4:

Research and analyze existing Python game engines and frameworks to determine if they can be used to optimize the game performance. Evaluate third-party libraries and tools that can be used to improve game performance, such as graphics libraries or caching solutions.

Week 5:

Synthesize the findings of the literature review and background study to develop a set of recommendations and solutions for improving the game performance.

2.2. Proposed solutions

The reported problem of a snake game in Python related to game performance and responsiveness, the timeline can vary depending on the available resources and the scope of the problem. Identify relevant academic and industry publications related to game development, Python programming, and performance optimization. Read and analyze the publications to gain a better understanding of the common performance issues in game development, best practices for Python programming, and techniques for performance optimization. Search online forums, communities, and blogs related to Python game development to gain insights and practical tips from developers who have encountered similar issues. Identify open-source projects or libraries that can be used to optimize Python game development. Investigate performance profiling tools and techniques to identify performance bottlenecks in the game code. Research and analyze existing Python game engines and frameworks to determine if they can be used to optimize the game performance. Evaluate third-party libraries and tools that can be used to improve game performance, such as graphics libraries or caching solutions. Develop and test the proposed solutions, such as implementing a better algorithm for collision detection or optimizing the game loop.Document the literature review and background study, including the sources used, key findings, proposed solutions, and testing results.

2.3. Bibliometric analysis

Bibliometric analysis is a method of analyzing and measuring scientific literature, primarily through citation analysis. It involves using quantitative methods to analyze patterns of publication and citation in a particular field or topic. Here is a possible outline for conducting a bibliometric analysis related to a snake game in Python:

Define search terms: Define the search terms related to snake game in Python, such as "snake game," "Python programming," and "game development."

Filter results: Filter the results based on relevance, such as filtering out irrelevant articles or those not related to game development.

Analyze citations: Analyze the citations of the selected articles to identify the most highly cited articles and authors in the field.

Identify trends: Identify trends in the publication of articles related to snake game in Python, such as the frequency of publications over time, the most common topics covered, and the most active research areas.

Analyze collaborations: Analyze collaborations between authors and institutions to identify the most active collaborators in the field.

Evaluate impact: Evaluate the impact of the most highly cited articles by analyzing the number of times they have been cited and the impact factor of the journals in which they were published.

Synthesize findings: Synthesize the findings of the bibliometric analysis to gain insights into the current state of research related to snake game in Python and to identify potential research gaps or opportunities for future research.

Document analysis: Document the bibliometric analysis, including the search terms, databases used, selection criteria, analysis methods, and key findings.

2.4. Review Summary

The literature review/background study related to snake game in Python focused on game performance and responsiveness. Many publications discussed common performance issues in game development, such as slow frame rates, long loading times, and stuttering animations. They also discussed various techniques for optimizing game performance, such as using a game engine or framework, implementing efficient algorithms, and minimizing the number of draw calls. Several publications provided best practices for Python programming, such as using optimized data structures, avoiding unnecessary memory allocation, and optimizing loops and conditionals. These techniques can be applied to Python code used in the game to improve its overall performance. The literature also covered various techniques for optimizing the performance of Python code, such as profiling and benchmarking, code optimization, and parallel processing. These techniques can be used to identify performance bottlenecks in the game code and to implement optimizations that improve game performance. Many publications discussed the benefits of collaboration and using open-source tools in game development. Collaborating with other developers can provide access to a wide range of skills and knowledge, while open-source tools can help reduce development time and costs. The literature review/background study provides valuable insights into the current state of research related to snake game in Python and offers several potential solutions for improving game performance and responsiveness.

2.5. Problem Definition

The problem in a snake game in Python can be defined as the need for the game to be designed in such a way that it provides a responsive and engaging experience for the player. This requires addressing several technical challenges, such as optimizing game performance ensuring smooth animations and implementing effective collision detection algorithms. The game must be designed with a user-friendly interface that allow players to easily control the snake and understand the game mechanics. The game should also have appropriate sound effects and visual cues to enhance the overall gaming experience. Another important aspect of the problem is the need to balance the game difficulty, ensuring that it is challenging enough to keep the player engaged, but not so difficult that it becomes frustrating. Achieving this balance requires careful consideration of the game mechanics, level design, and overall game flow. The problem in a snake game in Python is to create a game that is both technically sound and engaging for the player, providing a satisfying gaming experience.

2.6. Goals/Objectives

The overall goals/objectives of developing a snake game in Python are to create a technically sound game that provides an engaging and enjoyable experience for the player. To achieve this, the game must be optimized for smooth gameplay, responsive controls, and efficient collision detection. Visually appealing graphics and animations, appropriate sound effects, and user-friendly controls are also important to create an engaging user experience. Additionally, the game should include various features such as multiple levels, power-ups, high scores, and customizable game settings to enhance the overall gaming experience. Clear and concise documentation should also be provided to ensure the game is accessible to a wide range of players. Finally, using open-source tools and techniques and collaborating with the Python community will help create a community of developers who can share knowledge and resources and contribute to the overall advancement of game development in Python.

CHAPTER – 3

DESIGN FLOW / PROCESS

3.1. Evaluation & Selection of Specifications/Features

The evaluation and selection of specifications/features for a snake game in Python is an important process that can have a significant impact on the game's success. The evaluation process involves reviewing the game's requirements and identifying the most important features needed to meet those requirements. This can be done by prioritizing the features based on their impact on the overall user experience and the game's performance. For example, user-friendly controls, smooth gameplay, and visually appealing graphics are all important features that can greatly enhance the player's experience. Once the features have been identified, they can be evaluated based on their feasibility, development time, and impact on the game's performance. For example, if a feature is not feasible to implement or will take too long to develop, it may be necessary to prioritize another feature that can be implemented more quickly and easily. The features can be evaluated based on their impact on the game's user experience. The selection process involves choosing the features that are most important for the games success based on the evaluation criteria. This may involve making trade-offs between different features, based on their feasibility, development time, and impact on the overall user experience. For example, if adding multiple levels to the game will significantly increase development time and may not be feasible, it may be necessary to prioritize other features that can be implemented more quickly and still have a positive impact on the game's user experience. In addition to evaluating and selecting specific features for the game, it is important to consider how these features will work together as a cohesive whole. The game's design and mechanics should be carefully considered to ensure that they are intuitive and easy to understand for the player. The game should also be designed with scalability in mind, so that additional features and improvements can be added in the future. the evaluation and selection of specifications/features for a snake game in Python is a crucial step in the development process. By prioritizing features based on their impact on the user experience and game performance, evaluating feasibility and development time, making trade-offs, and considering the game's overall design, developers can create a game that is enjoyable and engaging for players.

3.2. Design Constraints

Design constraints are limitations that impact the development of a snake game in Python. These constraints can include technical limitations, such as hardware or software limitations, as well as limitations on time, budget, or resources. For example, technical constraints could include limited memory or processing power, which may impact the game's performance. Time constraints could include a deadline for the completion of the game, which may limit the amount of time available for development. Another design constraint is the need to balance the complexity of the game with its playability. A game that is too complex may be difficult for players to understand and enjoy, while a game that is too simplistic may not be engaging enough to hold a player's interest. The design of the game should aim to strike a balance between these two considerations. Another important

constraint is the need to ensure that the game is accessible to as many players as possible. This means considering factors such as user interface design, input methods, and compatibility with different platforms and devices. The game should be designed to be intuitive and easy to understand for players of all skill levels, and to work well on a variety of devices and platforms.
In addition to technical and playability constraints, the development of a snake game in Python must also consider legal and ethical constraints. These may include issues related to copyright and intellectual property, as well as concerns about inappropriate content or messaging. The game should be designed to comply with relevant laws and ethical standards. In summary, the design constraints for a snake game in Python include technical limitations, time and resource limitations, the need to balance complexity and playability, accessibility for a wide range of players and platforms, and legal and ethical considerations. By considering these constraints during the development process, developers can create a game that is engaging, playable, and compliant with relevant laws and ethical standards.

3.3. Analysis and Feature finalization subject to constraints

The analysis and feature finalization of a snake game in Python is subject to various constraints. These constraints may include technical limitations, time constraints, budget constraints, and ethical considerations. Technical constraints can limit the types of features that can be implemented in the game. For example, limited memory or processing power may restrict the complexity of the graphics or the number of game objects that can be present on the screen at one time. Similarly, the compatibility of different platforms and devices may constrain the types of input methods that can be used. Time constraints can also impact feature finalization. Developers may have a limited amount of time to complete the project, which means that some features may need to be prioritized over others. Budget constraints may also limit the amount of resources that can be allocated to the project, potentially impacting the scope of the game. In addition to technical and resource constraints, ethical considerations may also play a role in the analysis and feature finalization of a snake game in Python. For example, developers may need to consider whether certain types of content or messaging are appropriate for the intended audience. They may also need to consider issues related to privacy and data security. Despite these constraints, it is possible to create a successful and engaging snake game in Python by carefully analyzing the project requirements and prioritizing the most important features. Developers can work within the constraints to create a game that is both technically sound and enjoyable for players. By considering ethical considerations alongside technical and resource constraints, developers can ensure that the final product is both fun and responsible.

3.4. Design Flow

The design flow for a snake game in Python involves a series of steps that help developers create a functional and engaging game. The first step in the design flow is to define the requirements for the

game. This includes identifying the target audience, determining the game's objectives and gameplay mechanics, and specifying the technical requirements. Once the requirements have been defined, the next step is to create a design document. This document outlines the game's features, mechanics, and rules. It also includes information on the game's user interface, graphics, and sound effects. The design document serves as a roadmap for the development process and helps ensure that all team members are on the same page. After the design document is created, the next step is to begin developing the game. This typically involves creating a prototype or rough draft of the game, which allows developers to test and refine the gameplay mechanics. The prototype may include basic graphics and sound effects to help give a sense of what the final product will look and feel like. Once the prototype has been developed, the next step is to begin integrating the game's features and mechanics. This may involve coding the game's user interface, designing the graphics and sound effects, and implementing the game's rules and mechanics. Throughout this process, developers should continually test and refine the game to ensure that it is engaging and enjoyable for players. After the game has been developed, the next step is to test and debug it. This involves identifying and fixing any issues or bugs in the game, as well as optimizing the game's performance. Developers may also conduct focus groups or user testing to gather feedback from players and identify areas for improvement. Once the game has been tested and debugged, the final step is to release it to the public. This may involve publishing the game on a digital platform, such as a website or app store, or distributing it through other channels. After release, developers should continue to monitor the game's performance and gather feedback from players to identify areas for improvement.

Throughout the design flow, developers must also consider constraints such as time, budget, technical limitations, and ethical considerations. These constraints may impact the design decisions and features that are included in the game. However, by carefully managing these constraints and prioritizing the most important features, developers can create a successful and engaging snake game in Python. The controls should be intuitive and easy to learn, allowing players to quickly understand how to maneuver the snake and collect the food items. Developers may choose to use keyboard controls or touchscreen controls, depending on the platform and target audience. The graphics should be visually appealing and consistent with the game's overall theme and aesthetic. The colors, textures, and visual effects should all work together to create a cohesive and engaging experience for players. To help manage technical constraints, developers may use tools and frameworks such as Pygame. These tools provide a set of pre-built components and libraries that can be used to accelerate the development process and ensure that the game runs smoothly on a variety of platforms.