PAC-MAN Doodle

TEAM MEMBERS

SYED MUHAMMAD AHSAN (24K-2041)

SYED MUHAMMAD MUZAMMIL (24K-2000)

ANAS HAIDER (24k-2002)

Pac-Man: An Object-Oriented Game Project Report

Group Members

Syed Muhammad Muzammil – 24k-2000
Syed Muhammad Ahsan - 24k-2041
Anas Haider – 24k – 2002

Submission Date**:** 11 May 2025

1. Executive Summary

Overview

This project is a modern, object-oriented implementation of the classic arcade game **Pac-Man**. Developed in **C++** using **SFML**, it aims to recreate the nostalgic gameplay while showcasing core object-oriented programming (OOP) principles like encapsulation, inheritance, and polymorphism.

Key Findings

- Successfully applied OOP concepts for better code structure and modularity.
- Developed a playable, responsive version of Pac-Man with animated characters and interactive elements.
- Implemented basic ghost AI, scoring systems, and power-ups.
- Ensured the codebase is clean, maintainable, and extendable for future improvements.

2. Introduction

Background

Pac-Man, the arcade classic launched in 1980, remains an icon in the gaming industry. It involves navigating a maze, collecting pellets, and avoiding ghosts. Eating a power pellet enables Pac-Man to chase the ghosts for bonus points. This project revives the nostalgic experience with modern development tools using Object-Oriented Programming (OOP) in C++ and SFML for graphics.

Problem Statement

Classic games like Pac-Man often lack modular, reusable code structures, making enhancement and debugging challenging. This project addresses that gap by implementing the game using OOP principles to ensure clean, maintainable code.

Objectives
- Build a playable Pac-Man clone in C++ using SFML.
- Apply OOP concepts: encapsulation, inheritance, polymorphism, and abstraction.
- Develop game mechanics including player movement, collision detection, and ghost AI.
- Integrate power-ups and a scoring system.
- Use SFML for graphics, animation, and sound effects.
- Ensure a modular and extensible codebase.

3. Scope of Project

Inclusions
- Pac-Man Character: Controlled via arrow keys with collision detection.
- Ghost AI: Behaviors like chase, scatter, and frightened modes.
- Pellet System: Scoring based on pellet collection.
- Power-Ups: Abilities like ghost consumption and speed boosts.
- Maze Layout: Static, pre-defined maze environment.
- Scoring & Levels: Points-based progression with increasing difficulty.

Exclusions
- Complex AI algorithms beyond basic pathfinding.
- Multiplayer support.
- Custom maze creation or additional levels.
- 3D rendering or advanced graphical features.

4. Project Description

Research & Planning
- Studied core mechanics of Pac-Man.
- Defined an object-oriented class structure for characters, environment, and logic.
- Selected SFML for 2D graphics and event handling.

Game Mechanics Implementation
- Implemented movement and interactions for Pac-Man and ghosts.
- Collision detection with walls, pellets, and characters.
- Basic pathfinding for ghost movement.

Graphical Interface Development
- Integrated SFML for sprite rendering.
- Animated character movement using sprite sheets.

Game Logic & Features
- Scoring system, level transitions, and game-over scenarios.
- Power-ups altering ghost states and player speed.

Testing & Debugging
- Extensive testing for collisions, AI behavior, and performance.
- Fixed visual and logical bugs during development iterations.

Finalization
- Compiled code, completed documentation, and recorded demo.

5. Methodology

Development Approach
- Iterative Development: Milestone-based implementation.
- Agile Workflow: Regular updates and feature testing.
- OOP Design: Classes for Pac-Man, Ghost, Game Engine, Maze, and UI.
- Optimization: Ensured responsiveness and minimal resource usage.

Roles and Responsibilities
This project was completed as a group effort with three members, each contributing to different aspects of the development process:

-**Muzammil** handled the final evaluation and debugging phase. He reviewed the code for vulnerabilities and secure coding practices before submission. He was also responsible for preparing the project proposal and final report and presented the project during assessment.

-**Ahsan** worked on the SFML integration for the graphical user interface (GUI) and managed the connection between the game logic and the GUI. He also contributed to the implementation in the .cpp files.

-**Anas** focused primarily on the code development side, especially working with the .h files, contributing to the class structures and logic organization.

6. Project Implementation

Design and Structure
- Classes for Pac-Man, Ghost, Maze, Game Engine, and UI.
- Each entity encapsulated its behaviors and interactions.
- SFML used for rendering, input, and animations.

Functionalities Developed
- Pac-Man movement and collision detection
- Ghost AI (basic pathfinding and mode switching)
- Pellet and power-up system
- Score tracking and level progression
- Animated sprites and sound integration

Challenges Faced
- Balancing ghost AI behavior with game difficulty
- Managing sprite animations for movement
- Resolving issues with collision boundaries and event timing

7. Results

Project Outcomes
- Fully functional, playable Pac-Man clone
- Clean, well-documented C++ codebase using OOP
- Functional scoring, level, and power-up systems

Steps to Play the Game
- Open game terminal
- Type make in terminal
- Then write ./sfml-app
- Game runs

Testing and Validation
- Manual testing across all features
- Validated player and ghost interactions
- Confirmed scoring, power-ups, and transitions work as intended

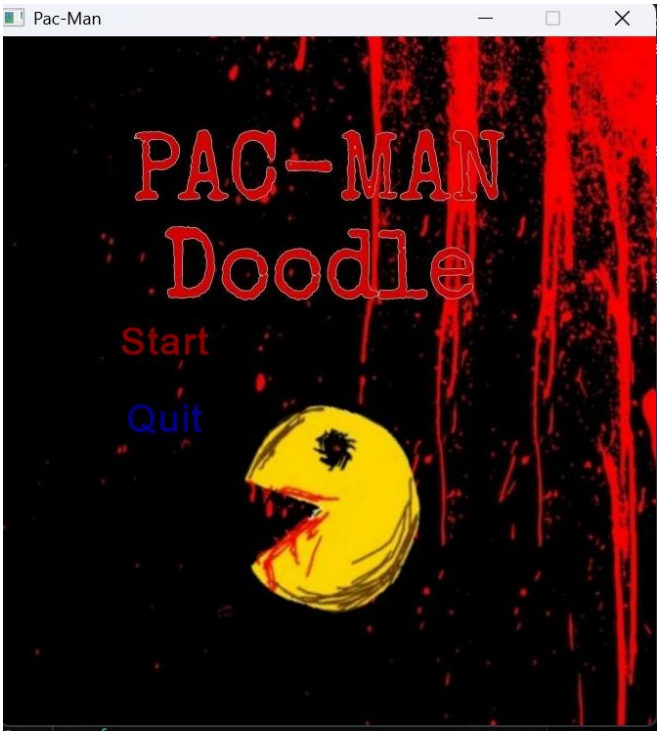
8. Conclusion

Summary of Findings
The project successfully showcases the effectiveness of OOP in game development. The modular structure enhances maintainability, while SFML provided a flexible foundation for rendering and interaction.

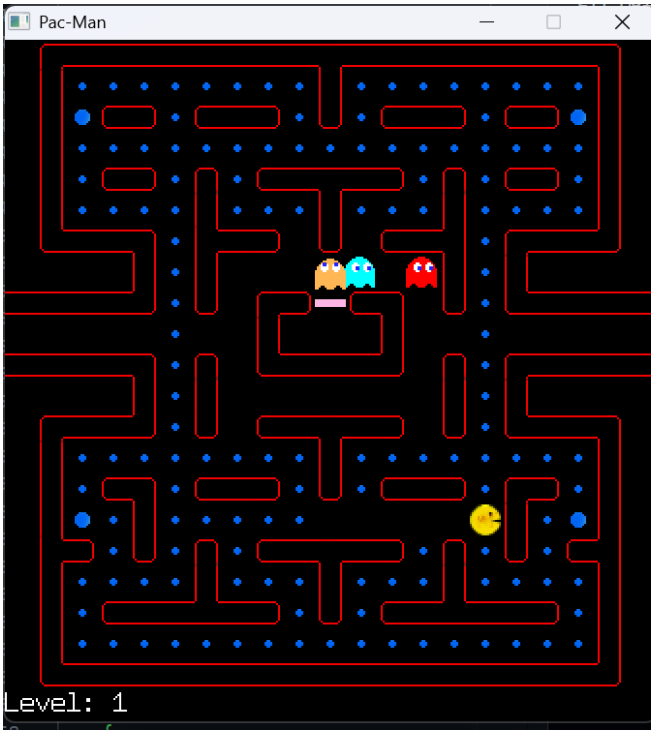
Final Remarks
This project deepened understanding of both OOP and game mechanics. It offers a solid base for future development, such as expanding ghost AI or adding new levels.

9. Project Demonstration

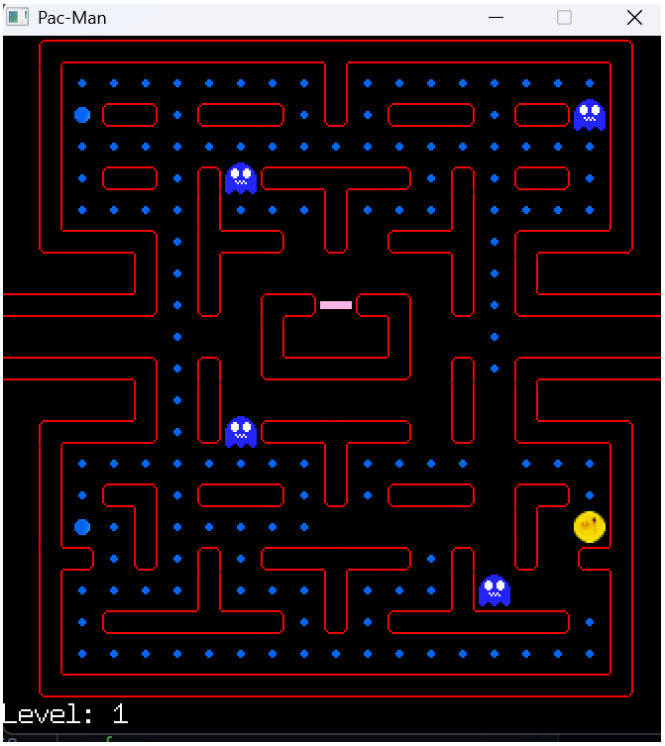
Start of Game:



Main Game:



After Eating fruit:



Ghost blinkers: