Name: Nguyen Khanh Toan - 104180605
Unit: COS30002 - AI for Games

Custom Project Plan

1. Introduction

This project involves creating a graph navigation problem and a GUI to demonstrate various pathfinding algorithms. The algorithms to be implemented include BFS, DFS, Greedy Best First Search, A* and Bi-directional versions of BFS and A*. The GUI will be interactive, allowing users to select algorithms, maps, or create their own maps. The program will follow an OOP structure, and UML diagrams will be used to demonstrate the design.

2. Problem Description

Pathfinding in graphs is a common problem in game development and AI applications. The AI system developed in this project will address the problem of finding the most efficient path from a starting node to a goal node in a graph. This is crucial for NPC navigation, puzzle solving, and other AI-driven functionalities in games.

3. Objectives

- Implement various pathfinding algorithms: BFS, DFS, Greedy Best First Search, A*, Bi-directional BFS, and Bi-directional A*.
- Develop an interactive GUI to demonstrate the pathfinding process.
- Allow users to choose different algorithms and maps or create custom maps.
- Design the program using OOP principles and represent the design with UML diagrams.
- Validate the algorithms and GUI through testing and user feedback.

4. Feature List

Algorithms:

- Breadth-First Search (BFS)
- Depth-First Search (DFS)
- Greedy Best First Search
- A* Search
- Bi-directional BFS
- Bi-directional A*
- Graph Navigation:

Program representation:

- Problem representation

- Node representation
- Graph creation and manipulation

GUI Features:

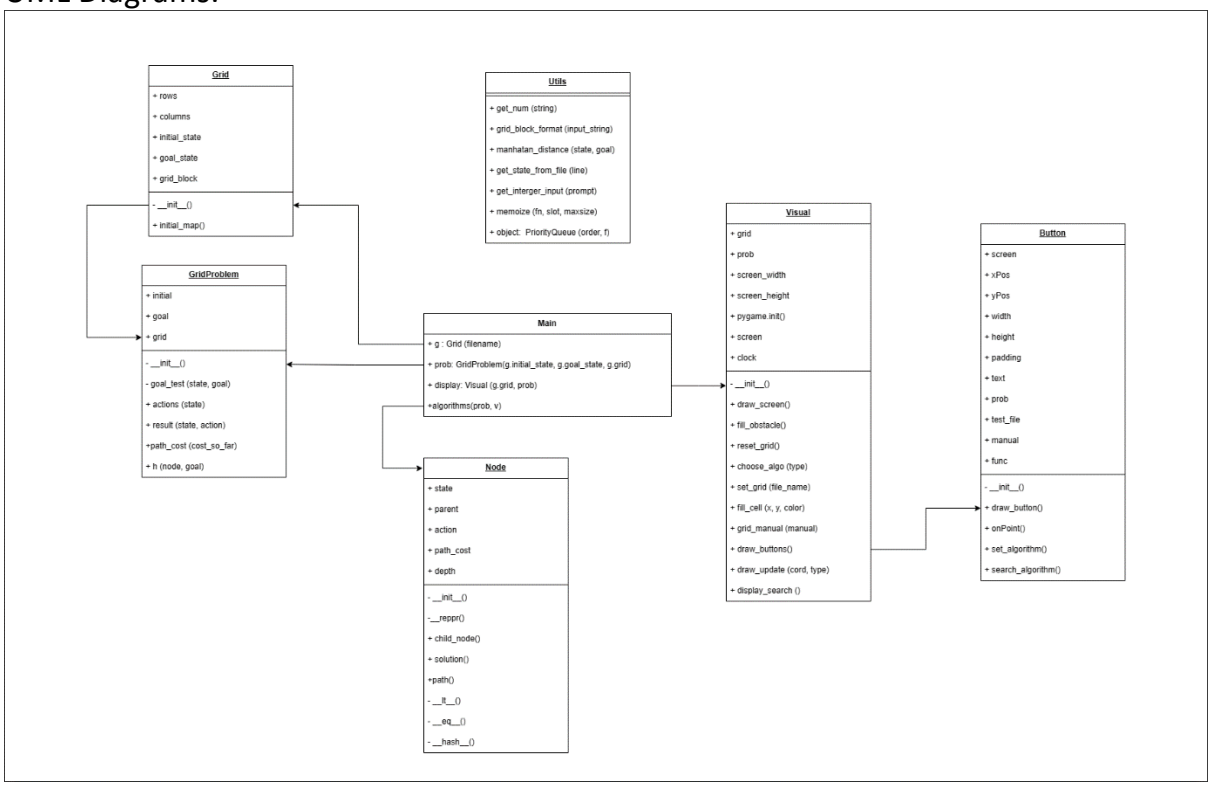
- Algorithm selection
- Predefined map selection
- Custom map creation
- Visualization of algorithm progress
- Interactive user inputs

OOP Structure:

- Class definitions for graph, nodes, and edges
- Separate classes for each algorithm
- GUI class for handling user interface

5.Architecture and Design

UML Diagrams:



Class Brief explanation:

- Grid Problem Class: Defines the problem for the graph search, including start and goal nodes, and the graph itself.
- Grid Class: Creates and manages the graph/grid environment.

- Node Class: Represents a node in the graph with essential attributes and methods.
- Visual Class: Handles the GUI drawing and visualization of the graph and the pathfinding process.
- Button Class: Represents buttons in the GUI with associated functionality.
- Main(): Containing Algorithms and handle users input.

6. Implementation Details

Development Tools:

Programming Language: Python
GUI Framework: PyGame
Library: PyGame, math, ultils,…
Document each release with version updates, new features, and bug fixes.

7. Conclusion

This project will provide a comprehensive exploration of pathfinding algorithms within an interactive GUI. The use of OOP principles and UML diagrams will ensure a well-structured and maintainable codebase, while thorough testing and validation will ensure a reliable and user-friendly application.