PACMAN PSEUDOCODE

# Project Description

Recreate the classic arcade game “PACMAN”

This is a single screen game in which the player controls a character while being pursued by a group of 3 computer controlled ghosts.

The game is set within a 2D maze, where the player moves using the four keyboard arrow keys . As PacMan moves through the maze, he will consume food, one per square of the maze. The objective is to clear the maze of all food without being touched by any of the ghosts.

# Game Components

## Pacman

Pacman is controlled by the player and can only move left to right, up and down. He can move anywhere within the maze, but must not pass through any of the walls throughtout. As he moves, he will pass over kibble objects which he will consume.

## Ghosts

Three ghosts will be controlled by the computer. They constantly move inside the maze but cannot pass through any walls within it. Their objective is to try to come into physical contact with pacman.

## Kibble

At the beginning of the game, a piece of kibble is placed within each square in the maze. PacMan consumes the kibble once he comes into contact with it. The kibble will disappear from the maze once consumed.

## Maze

The maze is the playing area of the game and consists of a series of walls that will block progress of both PacMan and the ghosts. Players must constantly navigate through the maze to be successful. The maze consists of a 20 x 20 grid, and remains in place through the entire game.

NOTE: playable maze area is 20 x 20, actual grid size is 22 x 22

## Point Scoring

The user scores points each time they consume a piece of kibble.

## Game Completion

The user wins the game when all kibble has been consumed.

## Game defeat

The player looses a life if they come into contact with a ghost. If they lose three lives, the game is over.

# Form Design

|  |  |
| --- | --- |
| Proposed Playing Area | |
| Lose Screen | Win Screen |

# Controls & Components

## Timer

Used to move set events forward at a predetermined interval.

## Player Interaction

The player will be able to control the horizontal and vertical movement of PacMan, but will be limited by the position of walls in thre maze.

## Computer Coltrol

The computer controls the three ghosts that are persuing PacMan. They will genrally move randomly aroiund the maze, but will move in the direction of pacman if he comes within their view.

The computer will also keep score, reset the playing area and determeine when the player wins, and when a player loses.

# Events

## Timer Tick

The game runs via a timer set at predetermined intervals. Each tick of the timer will run a method from the world class which runs all events within the game.

## Key inputs

The form will detect key inputs entered by the user to determine movement of the PacMan object

# First Iteration

## Minimum Viable Product

MVP will draw the playing area maze, along with the pacman object, and a ghoul object.

# Classes

## Controller Class

### Behaviours

Run the order of events for each tick of the timer

### Field Data

Pacman Object

Maze Object

Ghoul1 Object

## Maze Class

### Behaviours

Draws the maze which acts as the playing area of the game.

Updates the maze accordingly

### Field Data

Map String

Kibbles remaining

Wall Bitmap

Kibble bitmap

Blank square bitmap

## Creatures Class

### Behaviours

Parent class to ghouls and pacman objects

### Field Data

Maze

List to store animation frames

## PacMan Class

### Behaviours

Draws the pacman frames for animation

Determines the movement of pacman

Determines the position of pacMan within the maze

### Field Data

Position

Frames

## Ghoul1 Class

### Behaviours

Draws the ghoul frames for animation

Determines the movement of the ghoul

Determines the position of ghoul within the maze

### Field Data

Position

Frames

## MVP UML Diagram

Diagram

Description automatically generated

## MVP Sequence Diagram

Diagram

Description automatically generated with low confidence

## UML Diagram

Graphical user interface, application

Description automatically generated