Assignment Group Project for ECE3206

This assignment contributes 10% to course work marks. It consist of two questions.

Question 1 (4%)

In this question, you will work in-group (2 members) to create a hangman game. Use C++ string variable and object oriented principles. At least 2 classes must be used. Improve your game by giving a short hint related to the correct word. The due date to submit the solution to question 1 is **19 February 2021 before 5pm (week 13)**.

Rubrik for Assessment (8 Marks)

Each successful implementation of the following requirements earns a maximum of 2 marks. All Rubrik are demonstrated through **recorded video demonstration** of the developed program.

1. Program has at least 2 classes
2. Program is correctly implemented according to the sample output and work for multiple iterations.
3. Program able to give hint as image or text file. (image score higher mark)
4. UML class diagram with short report on program design and usage. Source code to be attached and subject to copy detection.

Question 2 (6%)

In this question, you will work in-group (2 members) to create a simplified pacman game. The rule of the original pacman game can be found in this link http://www.pacxon.net/pacman-rules.php and you can try play it online https://www.webpacman.com/ . Two versions of the program shall be delivered. The first version shall be used to evaluate the achievement in rubric 1 and 2. Student must use the given class declaration and main() driver function. For the 2nd version, students adapt the 1st version to improve it for showing the achievement of rubric 3.

The due date to submit the 1st version is **18 Jan 2021 week 9**. The due date to submit 2nd version is **19 February 2021 before 5pm (week 13)**. The demonstration and individual interview is 22 Feb.

Your program must declare and implement at least 3 classes, namely it follows the following classes. The maze must be read from a text file.

Class	**Responsibility**
Creature	Responsible for managing the pacman and monster in the game
Maze	Manage the maze.
CGame	Run the game Check if the movement is valid

Rubrik for Assessment (10 Marks)

Each successful implementation of the following requirements earns a maximum of 2 marks. Rubrik 1,2 and 3 are demonstrated through **recorded video demonstration** of the developed program.

1. Pacman able to move within the maze and obey the rules. Program able to accommodate different maze size.
2. Pacman able to eat the food and update its score
3. One monster moves in the maze and it can eat the Pacman
4. UML class diagram for the classes
5. Report with source code that explain how the game is played and designed. Source code will be intensively check for copying among students.

Rules for Pacman game

1. Pacman and monster must move within the boundary of the maze.
2. Pacman and monster cannot go over the maze wall.
3. Points are collected if the pacman goes over the maze location with food.
4. Game end if pacman get eaten by monster. Monster eat pacman if it go into the same position as pacman.
5. The program operation can be seen in the given sample program output

Driver function for version 1 program

```
#include <iostream>
#include <fstream>
#include <string>
#include "Game.h"

using namespace std;


int main() {
	CGame game1;
	Maze m1; // init and allocate memory
	m1.readMaze();
	m1.showMazeProperty();

	m1.drawMaze();
	Creature p(1, 3, 1); // starting position of pacman in maze at(1,3) where type=1 is pacman
	m1.placeCreatureInMaze(p);
	m1.drawMaze();

	bool status;
	char move='y';
	while (move != 'q') {
		m1.drawMaze();
		m1.showMazeProperty();
		p.showCurrentPoints();
		p.showCreaturePosition();
		cout << endl << "enter move >> ";
		cin >> move;
		// check if move is  valid in the maze
		game1.checkMove(move, m1, p); // if it is valid, update the pacman position and update maze
		m1.showMazeProperty();
```

```
        }


        m1.deleteMaze();
        system("pause");
        return 1;
}
```

Required class

```
class Creature {
        int row; // position of creature object (Pacman) in maze
        int col;
        int type; // 1 indicate pacman
        int points = 0;
public:
        Creature() {}
        Creature(int a_row, int a_col , int a_type) {
                row = a_row;
                col = a_col;
                type = a_type;
        }
        void getPosition(int& a_row, int& a_col); // get pacman position in maze and store in a_row, a_col
        void showCreaturePosition();
        void setPosition(int a_row, int a_col);
        void updateCreaturePosition(int nrow, int ncol);// update based on new position nrow, ncol
        void addPoints(int a_point); // add points collected
        void showCurrentPoints(); // show collected reward points
};


class Maze {
        int numRow;
        int numCol;
        int** mat; // 2D matrix that stores the maze

public:
        Maze() {}
        void showMazeProperty(); // display the maze 2D matrix stored in mat
        void readMaze(); // read maze from text file and populate mat
        void drawMaze();
        void deleteMaze(); // release memory
        // update maze based on location of pacman object p and its previous location
        void updateMaze(Creature p, int row_, int col_);
        bool isValidMove(int nrow, int ncol); // Check if the move to new location (nrow,ncol) is valid
        void placeCreatureInMaze(Creature p); // Position pacman object p into the maze
        // return points as reward if move into food at location (r,c)
        int getReward(Maze a_maze, int r, int c);
};

class CGame {
public:
        // check if the move by creature is valid or not
        void checkMove(char move, Maze& a_maze , Creature& a_creature);

};
```