PACMAN REVISITED

Do you remember the old game of PACMAN? If not now you will. It's time now for you people to wear your thinking caps and make a code for your own PACMAN Player. On the occasion of I-fest 2011, Eureka this time we present to you "PACMAN REVISITED". The rules have been tweaked a little in order to befit the situation. So here we go on with the instructions and guide to PACMAN REVISITED.

CONTENTS:

The folder here consists of 5 .CPP files. The description is as followed:

GameEngine.cpp: This file is the controller of the entire game. The code in this file doesn't really matters to you. You just need to Compile & Run this file to check the working of your player. Still we are providing the comments with the code just in case if you feel like looking into it.

Ghosts.cpp: This file contains the code for ghosts (the old enemies). Again this is none of your concern. This code controls the ghosts present in the map.

player00.cpp: This is a sample player provided by us. This is a very basic player made just to show you how to code for your player. You can always use this player to compete against your own player, but keep in mind this is a very basic player so winning with him doesn't guarantees anything☺.

playerXX.cpp: This is the thing you have to code for yourself. Here the XX has to be replaced by your team number and after completing its code you have to send it to us. If your team number is 8 then you have to replace XX by 08 and if its 14 the replace XX by just 14.

gameGUI.cpp: This file contains all the functions related to GUI display and the code in this file does not really matter to you. Just ignore it.

map1.txt: This is a text file for one the map that will be used in final showdown. Yes that's right this just one out of the three maps you player will be fighting on. Here:

- A '#' denotes wall. Your player will not be allowed to pass through them.
- A 'o' (small o) denotes a coin in the field. (You have to concentrate on this thing ☺)

Other than this once when the game will start you will see some more things on the console. They are:

- A '.' will represent a place from where a coin has already been lifted. You can move to such a location but you will not be able to gather any coin from there.
- '1' will denote position of first player and '2' will denote the position of second player.
- 'G' will denote position of ghosts on the field.

Things that matters:

There is a structure called Players:

```
struct Players
{
   int x,y;
   int direction;
   int health;
   int points;
};
```

Here (x,y) are the coordinates of the player, direction will denote the last movement of the player i.e. how the player reached his current coordinates.

For Directions we have five pre-processor directives namely
#define STAY 0
#define UP 1
#define DOWN -1
#define RIGHT 2
#define LEFT -2
As the name suggests 'STAY' denotes no movement, 'UP' denotes an upward movement, 'DOWN' denotes a downward movement, 'RIGHT' denotes a movement in the right direction and 'LEFT' denotes movement in the left direction.

Health denotes the total health left of the player. In this simulation we are providing 3 Units of maximum health to each player. Your health decreases in two scenarios:

- If you get caught by a ghost. You will lose one health in this case.
- Or you have a collision with the player in the field. In this case both will lose one health.

I f your health reaches 0 then you will not be able to play anymore. Remember in any case ghosts never leave the field.

The Ghosts class also have its members as x, y and direction.

Other Pre-Processors are:
#define ROWS 19
#define COLS 43
#define N_GHOSTS 3
#define MAX_HEALTH 3
#define MAX_MOVES ROWS*COLS
#define ENABLEGUI 1

Here ROWS and COLS denote the dimension of the field. N_GHOSTS denotes the number of Ghosts present on the field. MAX_HEALTH is the maximum initial health given to each player. MAX_MOVES denotes the maximum number of moves that will played in the game.

The game will stop if both the player gets out of the game or MAX_MOVES have been made or all the coins have been consumed or if you try to breach Arena walls.

YOUR JOB:

You have to complete the code for your player i.e. playerXX.cpp specially the Move function.
The Game function in the GameEngine.cpp makes calls to Move function of both the player and accordingly validates the moves and looks for any collision/crossovers.
Whenever the Move function of your player is called you are supposed to return one of the 5 direction values i.e. STAY, UP, DOWN, RIGHT or LEFT.
To help making your decision the function provides you with following arguments:
int map[ROWS][COLS]: It contains the integer format of the map that is in consideration. Here:

- A wall is represented by a value -1.
- A coin is represented by a value 1.
- A normal location is represented by a value 0.

const Players myPlayer: This is the information about your player (the entire struct described above).
const Players opponent: This is the information about your opponent's player (again the entire struct described above).
const Ghosts g[N_GHOSTS]: This array contains the information about all the N_GHOSTS number of ghosts present on the field.

If you try to change the information about your own player, opponent player, the map or the ghosts your player won't be able to play anymore.
The winner will be the one who collects the maximum coins out of the map. And the overall winner will be the one with maximum wins out of three matches (in 3 different maps).

Moreover the ghosts provided here are random moving ghosts but in the original game there will be two ghosts who will be a little more intelligent and can attack a player by looking at their position.

For further details mail your queries to ifest.ca@gmail.com .
--
Abhay Gupta
Ph: +91-9997064805