

## System Programming (MEEC/MEAer)

### Project Assignment 2019/2020

In this project the students will implement a simple distributed pacman game:

- (<https://en.wikipedia.org/wiki/Pac-Man>).

The game is composed of a board where monsters and packmen move. The board will have bricks that restrict the movement of the characters, and fruits that give the characters extra power.

The architecture of the game will be client-server. One server handles the board and implements all the game rules, while the multiple clients allow users to play.

## 1 Architecture

The architecture of the game is pure client-server as described in the figure. The server accepts connection (using INET stream sockets) from the clients, and manages the board (characters and fruits). Every time the user changes the position of one of his characters the client sends a message to the server then, the server verifies if the movement is correct, updates the position of such character, update the stat of the game and send updates to the clients. The clients when receive and update from the server change the graphical window to make it consistent with the game board.

### 1.1 Server

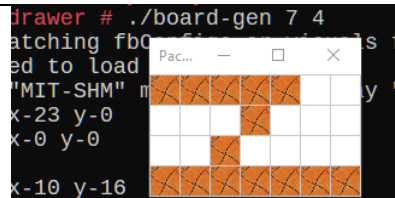
The server will load a board with a certain size and bricks' locations from a file, creates a SDL graphical window and creates a socket that will receive connections.

The definition of the board is provided by a file that is loaded in the start of the server. This file contains the size of the board (`n_cols` and `n_lines`) and the position of all the bricks.

The structure of this file is as follow:

- First line contain the number of columns (`n_cols`) followed by the number of lines (`n_lines`) of the board.
- For each line on the board the file will contain one line with `n_col` characters.
- An empty place in the board is represented by a space (' '), while a brick is represented by 'B'.

A board generator is provided. The board generator receives as arguments the number of columns and the number of lines, allows the user to place bricks (the left mouse button places a brick in place, the right mouse button removes a brick) and saves a **board.txt** file.

Usage	board.txt
	<pre>7 4 BBBBB \n   B  \n   B  \n BBBBBB</pre>

The server should assign to each client a unique PlayerID and store the relevant player/client information in a list or array.

The server will be responsible for verifying all received messages, update the board in accordance with the game rules, and communicate any update on the board to all the connected clients.

The server sends every minute the game score board to all clients.

## 1.2 Client

The client receives from the argv the address (IP address and port) of the server and the color of the characters (pacman and monster).

The client connects to the server, sends the color of the characters and receives (not necessarily by this order):

- the size of the board,
- the location of this player characters
- location of all the bricks, fruits and all the players.

The client creates a SDL2 windows that present the board and allows the user to control the characters.

During the game the client sends the position or movement of the characters and receives the movements of the other players' characters.

When the server sends the game score board, the client should print it using **printfs** in the console.

## 1.3 Communication protocols

Students should define the communication protocols for:

- Establishing of the connection (with exchange of all necessary information)
- Regular game play
  - sending of movement/new positions from client to the server
  - update of positions from server to clients)
- Game score board
- Client disconnect

These protocols should define every message structure, order and sequence.

All messages exchanged should follow the previously defined protocols, but at reception the message structure and data validity should be verified.

## 2 Game rules

The rules of the game should be implemented by the server as follows.

### 2.1 Player characters

Each player/client will control a monster and a pacman.

The color of these characters is defined by the player.

Two players can have characters of the same color.

### 2.2 Game duration

A game starts when the server is initialized and only ends when the server terminates.

### 2.3 Number of players

The minimum number of players is zero.

The maximum number of players depends on the available places in the board.

When a client connects, if it is impossible to place the two new characters in the board (because de board

is full) the client is kicked out.

## 2.4 Client connection

When the client connects to the sever, the server assigns to the characters two random positions on the board.

## 2.5 Client disconnection

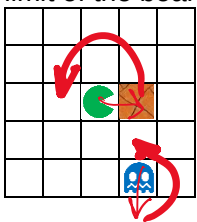
When the client disconnects his characters are erased from the board.

## 2.6 Character movement

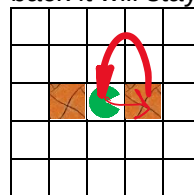
The characters can only move to the 4 adjacent places (down, up, left, right).

A character can only move two places per second.

If a character goes against a brick or the limit of the board it will get bounced back.

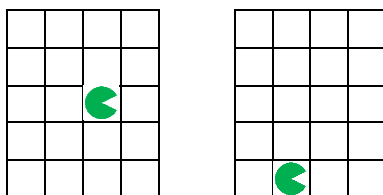


If a character cannot be bounced back it will stay at the same place



## 2.7 Character inactivity

If a character is inactive (not moving) for 30 seconds it will be placed at a random position, and the inactivity counter is reset.



## 2.8 Fruits

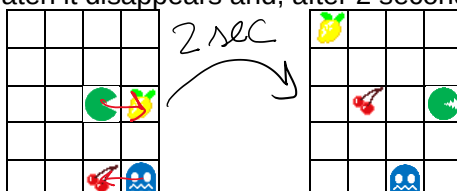
At every moment there should be  $(n\_players - 1) \times 2$  fruits (either lemons or cherries):

- 1 player – zero fruits, 2 players – two fruits, 3 players – four fruits, ...
- When there is only a player in the board all fruits should be removed

The fruits are placed at random positions.

Characters (pacmen and monster) eat fruits every time they go into a place that contains a fruit.

When a fruit is eaten it disappears and, after 2 seconds, is replaced by a new one on a random position.



## 2.9 Superpowered pacmen

When a Pacman eats a fruit, the pacman gets superpowered. A superpowered pacman has teeth: 🍷

After a superpowered pacman eats two monster he becomes a normal pacman.

## 2.10 Superpowered monsters

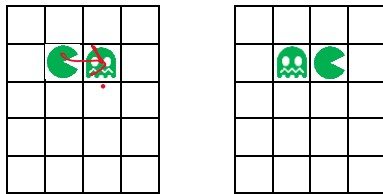
Monsters do not get superpowered.

When a monster eats a fruit nothing happens to the monster.

## 2.11 Characters interaction

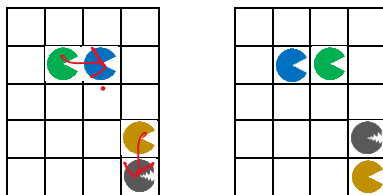
When a character moves into a place occupied by another character the following happens.

### 2.11.1 Characters of the same player



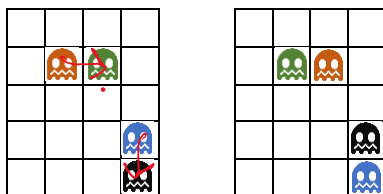
The characters change positions.

### 2.11.2 Pacman moves into pacman



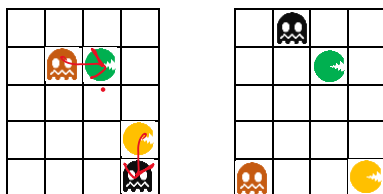
The characters change positions.

### 2.11.3 Monster moves into Monster



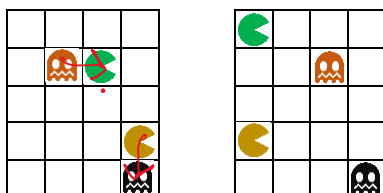
The characters change positions.

### 2.11.4 Monster against Superpowered pacman



The monster is eaten, and is moved to a random position.

### 2.11.5 Monster against normal pacman



The pacman is eaten, and is moved to a random position.

## 2.12 Game score board

Every player/client has one counter of eaten things (fruits and characters) by his packman and monster.

Every time a character eats something the corresponding client/player counter is incremented.

The score board is sent every minute to all the connected clients.

When a client disconnects he is removed from the scoreboard.

When there is only one player/client left, his counter is reset to zero.

### **3 User interface**


#### **3.1 Display**

The server and all clients have a SDL Windows that shows the board (with all the characters, bricks and fruits) updated in real time. Every time the server updates a position of a character or fruit, this change should be made in all clients windows.

#### **3.2 Characters control**

The clients can control his pacman and monster in the SDL Window.

The pacman is controlled by the mouse.

The Monster is controlled by the keyboard arrow keys: 

#### **3.3 Game score board**

The game score board is printed in each client console.