Computer graphics and multimedia
UDL MINF 20-21

Package 1

Danillo Lange Souza Borges dos Santos

Task 1. Random generation of a Pacman map

a. A constructor procedure that receives two integer input parameters (number of rows and number of columns).

```
// x and y are the desired size of rows x columns entered by the user
void mazeGenerate(int x, int y) {
		initializeGrid(x, y);
		step(1,1);
		addRoom();
		mapMirror();
		printMaze();
}

void initializeGrid(int x, int y) {

		maze_width = x;
		maze_heigth = y;

		maze_grid = new char [(int)maze_width*(int)maze_heigth];

		for (int i = 0; i < maze_width * maze_heigth; i++) maze_grid[i]= '#';
}
```

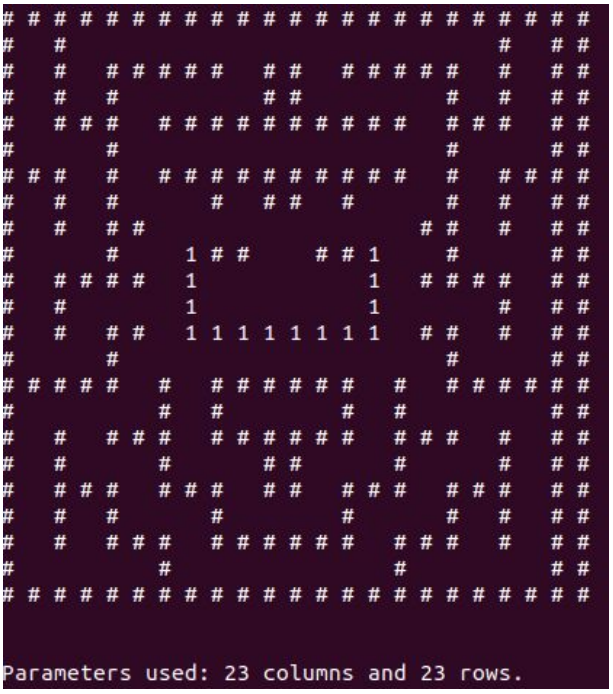
b. A random generation procedure that sets the value of each square to “wall” or “corridor” satisfying the previously mentioned requirements.

```
char step(int x, int y) {
		int direction[4];
		direction[0] = NORTH;
		direction[1] = EAST;
		direction[2] = SOUTH;
		direction[3] = WEST;

		// shuffle the desired direction
		for (int i=0; i<4; ++i) {
			int r = rand() & 3;
			int temp = direction[r];
			direction[r] = direction[i];
			direction[i] = temp;
		}

		maze_grid[translateXY(x,y)] = ' ';

		// Loop to attempt to visit that direction
		for (int i=0 ; i<4 ; i++){
```

```
                    //Initialize aux variables
                    int dx=0, dy=0;


                    switch (direction[i])
                    {
                            case NORTH: dy = -1; break;
                            case SOUTH: dy = 1; break;
                            case EAST: dx = 1; break;
                            case WEST: dx = -1; break;
                    }

                    // Aux variables to offset
                    int x2 = x + (dx<<1);
                    int y2 = y + (dy<<1);

                    if (checkBounds(x2, y2)) {
                            if (maze_grid[translateXY(x2, y2)] == '#') {
                                    maze_grid[translateXY(x2-dx, y2-dy)] = ' ';

                                    // repeat recursively
                                    step(x2, y2);
                            }
                    }

            }
            return *maze_grid;
}
```

c. A procedure for printing the map in text mode.

```
void printMaze() {
        for (int y = 0 ; y < maze_heigth; y++) {
                for (int x = 0; x < maze_width; x++) {
                cout << maze_grid[translateXY(x,y)] << " ";
                 }

                cout << endl;
        }
        cout << endl;
}
```

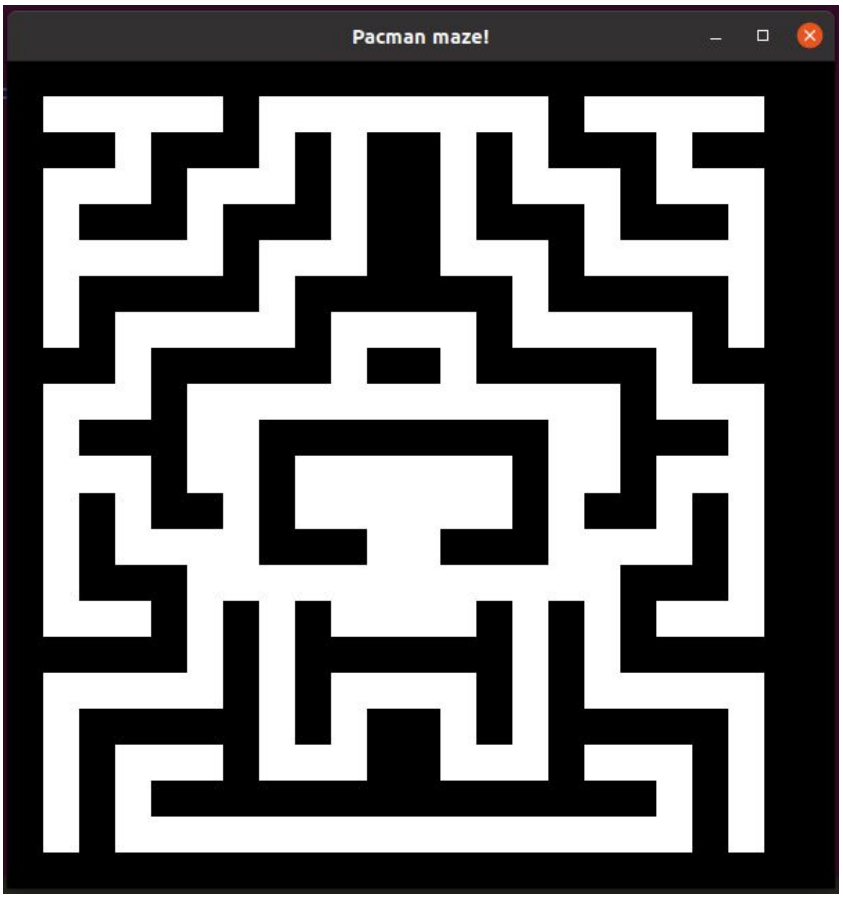


Task 2. Draw the Pacman map on a graphical window

Code:

```
void display() {
  glClearColor(0.0,0.0,0.0,0.0);
  glClear(GL_COLOR_BUFFER_BIT);

        for (int x = 0; x < maze.maze_width; x++) {
         for (int y = 0; y < maze.maze_heigth; y++) {
                if (maze.maze_grid[maze.translateXY(x,y)] == '#' || maze.maze_grid[maze.translateXY(x,y)] == '1') {
                        glColor3f(0.0,0.0,0.0); // Black

                }
                else if (maze.maze_grid[maze.translateXY(x,y)] == ' ')
                        glColor3f(255.0,255.0,255.0); // White

                glBegin(GL_QUADS);
                        glVertex2i(x * (cell_width), y * (cell_width)); // top left
                        glVertex2i(x * (cell_width) + cell_width, y * (cell_width)); // top right
                        glVertex2i(x * (cell_width) + cell_width, y * (cell_width) + cell_width); // bottom right
                        glVertex2i(x * (cell_width), y * (cell_width) + cell_width); // bottom left
                glEnd();
        }
  }
  glutSwapBuffers();
}
```

Result:



Bugs to be fixed:

- Invert printing in OpenGL;
- Debug all possible sizes entered for the maze
- Right wall too thick

Included code in activity:

Maze.cpp

main.cpp

Makefile