Project 2 – CSci 115 Algorithms & Data structures

Department of Computer Science, College of Science and Mathematics

Deadline: Wednesday, May 9th, 2018

Programming language: C++

Group project: Yes (max 2 students)

Learning outcomes:

- Software development
- Object Oriented Programming with C++
- Data structures: matrix, graph, shortest path algorithm
- Group project

The goal of this project is to create a turn-based video game. It is a 2D game where the player has a bow, and must reach the chest in the maze, there are enemies in the maze, the player can find arrows in the maze to shoot at enemies.

During the labs, you will get information about how to display an image. The implementation related to this aspect should not prevent you to start the implementation by creating the different data structures and classes related to the project.

- We consider a maze represented as a **matrix of size n x n**. You can select an appropriate size so you can place enough elements in the game. The maze contains different types of cells:
    - 0: empty (where it is possible to move)
    - 1: full (a wall)
    - From 20 to 29: enemies (for 10 enemies maximum)
    - 3: the player
    - 4: the chest to find (goal of the game)
    - 5: bags of arrows.
- At each turn:
    - The player can move one case (up/down/left/right) or shoot an arrow if he has some arrows.
        - You cannot traverse the walls
        - If you go inside an enemy: you die.
        - If you go in the chest, you win.
        - If you go to a bag of arrows, you win 1 arrow.
        - You can only shoot at an enemy if there is no wall between the player and an enemy (there is not restriction of distances between the player and the enemy).
    - The enemies in the maze will go towards the player by taking the **shortest path** in the maze.
        - It is not possible to have two enemies on the same cell.
        - Enemies can walk over the chest and bag of arrows.
        - If an enemy goes on the cell of the player: the player dies.
- The game ends when all the enemies are eliminated or if the player has found the chest or an enemy has reached the position of the player.

The maze can be stored in a text file where each character can represent a type of cell in the matrix.

The graphical user interface can be represented with buttons.

You are strongly encouraged to add additional functionalities.

Provisional marking scheme:

1. The project is submitted with a readme file that explains what it contains, who has done the project, and it tells how to use the files.
2. The project compiles.
3. The project is commented (e.g., head of the functions)
4. The project can run and it is possible to test the game with an appropriate level of difficulty
5. The project contains classes.
    1. A class for the Maze
    2. A class Player
    3. A class Enemy
6. Functions
    1. Display a maze based on the values from a matrix
    2. Collisions related to the actions of players
    3. Collisions related to the actions of the enemies
    4. Shortest path function for the enemies
    5. Determine if the level is completed
    6. User control
        - Move up/down/left/right
        - Shoot arrow up/down/left/right
          (you may use 4 keys + 1 button to toggle between move and shoot)
7. A 1-page report to explain the gameplay and how to the play the game.
8. Effort for the creation of the maze
9. Effort for the use of graphical elements that properly represents the different elements in the game.
10. A menu to start the game or exit the program, after each level, you may go back to the menu or to the next level.
11. Possibility to have multiple levels in the game.
12. Possibility to load levels from a file.