Project CSci115 Spring 2019: Turn-based Stealth Game
(group project – 2 students maximum)

Deadline: Week 15 (Last day of instruction)

Programming language: C++

Group project: Yes (max 2 students)

Learning outcomes:

- Software development
- Object Oriented Programming with C++
- Data structures: arrays, lists, 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 to reach a chest in a maze, there are smart enemies in the maze, the player must avoid the enemies to reach the chest.

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.

- It is a 2D game where the player has to find a chest in a maze. There are enemies in the maze. These enemies have a direction. They can only detect the player if they can see it. The goal is to avoid the enemies.
- The enemies have the following properties:
    - An enemy will go towards the player **only** if it can see the player, i.e. if there is no wall between the enemy and the player, in one of the 4 directions (up/down/left/right). The enemy will stop chasing the player if it cannot see it anymore **and** the distance between the player and the enemy > 5)
    - An enemy will move in the maze and will change his direction randomly only at the intersection, he will go back on his steps only at a dead end.
- We consider a maze represented as a **matrix**. The maze contains different types of cells: empty A (where it is possible to move for the players and the enemies), empty B (bushes) (where it is possible to move for only the player), full (a wall), enemies, the player.
    - There are places where the enemies cannot go (bushes) and where the player can be safe.
- At each turn:
    - The player can move one case (up/down/left/right).
    - It is not possible to have 2 enemies on the same cell.
- The game ends when the player has found the chest or an enemy has reached the position of the player (an enemy caught the player).
- The maze can be stored in a text file where each character represents a type of cell in the matrix.

- The graphical user interface can be represented with buttons to move the character up/down/left/right/next turn.

Code constraint

Your code must not contain any **break** or **continue** inside loops.

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 runs and it is possible to test the game with an appropriate level of difficulty to complete a level and understand the behavior of the enemies.
5. The project contains classes.
    1. A class for the Maze
    2. A class Player
    3. A class Enemy
6. Main 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
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.
13. Personal touch to improve the project, by adding relevant functionalities.