CS102 – Algorithms and Programming II
Programming Assignment 1
Fall 2025-2026

ATTENTION:

- Compress all of the Java program source files (.java) files into a single zip file.
- The name of the zip file should follow the below convention, where you replace the variables "Sec1", "Surname" and "Name" with your actual section, surname and name:

  CS102_Sec1_Asst1_Surname_Name.zip
- Upload the above zip file to Moodle by the deadline (if not, significant points will be taken off). You will get a chance to update and improve your solution by consulting to the TAs and tutors during the lab. You have to make the preliminary submission before the lab day. After the TA checks your work in the lab, you will make your final submission. Even if your code does not change in between these two versions, you should make two submissions for each assignment.

GRADING WARNING:

- Please read the grading criteria provided on Moodle. The work must be done individually. Code sharing is strictly forbidden. We are using sophisticated tools to check the code similarities. The Honor Code specifies what you can and cannot do. Breaking the rules will result in disciplinary action.

SIMPLE BOARD GAME IMPLEMENTATION

For this lab assignment, you will implement a Java console application for a simple board game for 2-4 players. The game takes place on an 11 by 11 square grid with players making a move on their turn. The game board also contains unpassable blocks and objects that give points to the player when picked up. Each player controls one character that moves like the rook in chess; that is, it can move any number of cells horizontally or vertically. If the movement stops on a cell occupied by an opponent character, the opponent's piece is captured, and the capturing player gains 5 points. The captured opponent can no longer continue the game. If the movement ends on a cell with a pickable object, the player picks up the object and receives 2 points. The picked object disappears from the board. The players cannot pass through blocks, opponents, or other player characters.

At the beginning of the game, your program should ask for the number of players, with the options being 2, 3, or 4. Then, allow each player to enter their name. The game decides on a random order for the players. Each player enters a direction (West, East, North, or South) and the number of cells to move along that direction. The game should check if the entered direction and number of cells are valid; if not, it should ask the player to enter correct values. Each player has to make a move on their turn; they cannot skip their turn. The game ends when only one player is left or all the pickable objects are acquired. When one of these conditions is true, display each player's points and declare the one with the highest amount as the winner. Note that the players who remain on the board may not win the game if their total points are not the highest.

Each player starts from one of the corners of the board: The first player starts from the top-left, the second player starts from the top right, the third player starts from the bottom-left, and the fourth player starts from the bottom-right cell. The board contains an unpassable block between each horizontally or vertically aligned character, so they cannot capture their opponents directly on the first turn. The board also contains at least five randomly positioned blocks. These randomly positioned blocks should not be adjacent to any existing block or player starting location, whether horizontally, vertically, or diagonally.

You will also position at least 10 pickable objects on the board randomly. These pickable objects cannot be positioned into already occupied locations, such as the player starting locations or the cells that contain a block; they also cannot be adjacent to the player starting locations; however, pickable objects can be adjacent to other pickable objects. Each object is one or two points; this information is hidden from the player unless they pick up the objects. The objects appear the same on the board until a player picks them up. When the object is picked up, the program must display a message about which player acquired how many points. Similarly, you should display a relevant message when a player captures another player.

At the beginning of each turn, display the current points of each player, and then the current status of the map. You will use the numbers to indicate player positions, '.' to indicate unoccupied cells, "X" to indicate unpassable blocks, and "*" to indicate pickable objects. For example, a starting map can appear as follows:

```
1.*..X....2
........X..
...X.......
X.......X..
..*...**...
X......*..X
....X.*....
*..........
X*..*...X..
....*......
3.*..X....4
```

During a player's turn, ask for a direction and the number of cells to move along that direction. The entered combination should not pass through any block or opponent. To capture an opponent, the player should end their movement on top of the captured piece, just like in chess. To pick up an object, the player should end their movement on top of it. Unlike blocks or opponent players, a player can pass through pickable objects; in this case, they are not picked up. For example, for the given map, the first player can move three units to the East and skip the point object; however, if the first player moves 1 unit to the East, s/he picks up the point object, receiving either one or two points based on its worth.

Preliminary Submission: You will submit an early version of your solution before the final submission. This version should at least include the following:

- The methods to generate and display the map should be complete.
- The logic for moving the player should be complete.

Not completing the preliminary submission on time results in 50% reduction of this assignment's final grade.