Assignment IP.1

New Attempt

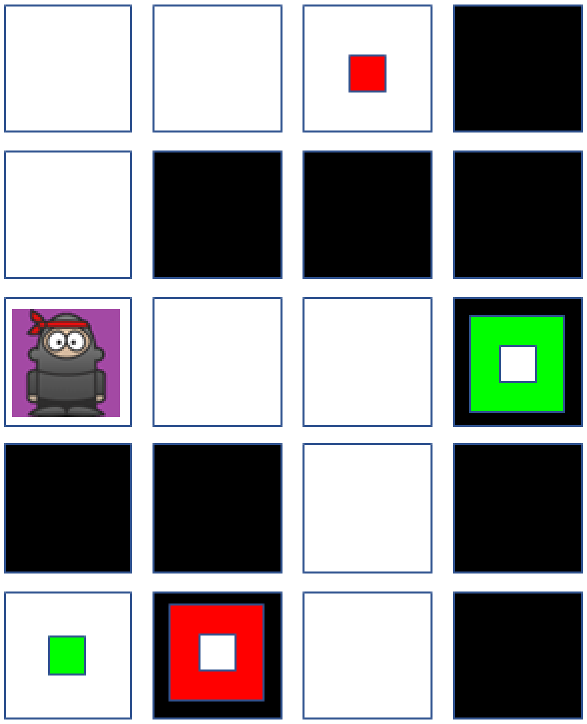
Due Nov 2 by 1pm **Points** 130 **Submitting** a file upload
Available after Oct 25 at 12pm

Individual Project CS 3733

Your task is to develop a small standalone application in Java to present Escape Room scenarios to be solved. Here is our hero, Ninja-se:

Ninja-Se

Ninja-se is scared! He needs to escape from the following 4 x 5 room with 20 cells arranged in five rows and four columns. Ninja-se can move from one cell to another, but only to the cell up, down, left, or right from his current location. Some cells (colored BLACK) are walls and prevent movement. Some of the cells in each room contain a colored key, shown as small colored squares in the image below. There will be a corresponding locked wall for each key with matching color. If Ninja-se is in a cell with a key he can pick it up but he can only hold one key at a time, so requesting to pick up a new key will automatically drop the key he had been holding.



To escape a room, Ninja-se must unlock all of the locked doors. After some thought, Ninja-se makes the following moves.

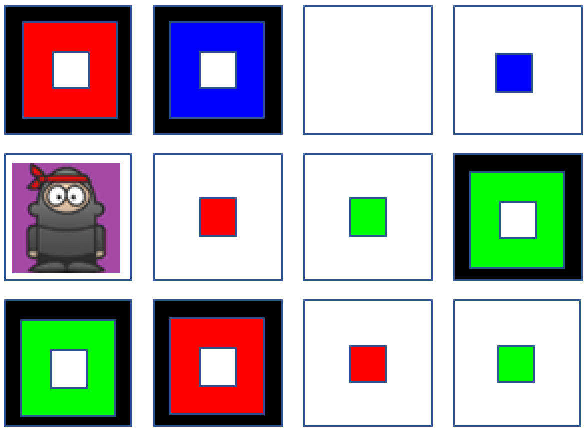
1. Move up and over in four moves and then pick up the red key.
2. Then in nine moves head all the way back down to the locked red wall on the bottom row which opens it up.
3. Then move left and pick up the green key
4. Then in five moves unlock the locked green wall in the right-most column.
5. Ninja-se is then FREE!

In a valid configuration, there will always be a single matching key for each locked wall, and each locked wall must be opened.

These escape rooms have **up to 8** rows and columns and can become complicated, even in small configurations. There are some observations:

- Ninja-se can always move into an empty cell.
- Ninja-se can only hold one key at a time. Once a key is used to unlock a wall, the key and the locked wall vanish and Ninja-se is free to pick up another key.
- If Ninja-se is holding a key and tries to pick up another key, Ninja-se drops the key he is holding and picks up the requested key.

Can you find a way to escape the following room? At first I thought 23 moves, then I found a 21 move solution. Can you do better? I think you can!



In your final application, the player can choose to play any one of three pre-defined levels, each one having a different rectangular shape and configuration of obstacles. **By default, the game starts in level 1**. When starting a level, Ninja-se and all keys are placed in their designated cells and all locked walls will be locked.

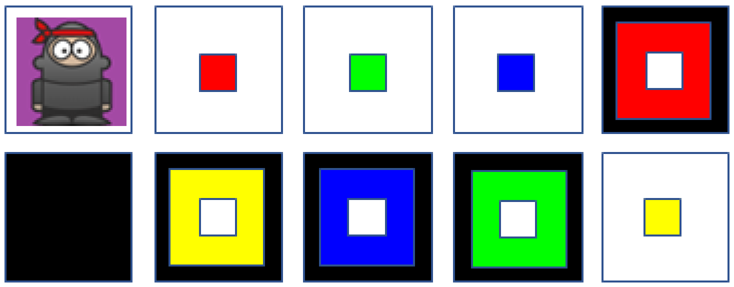
The player can choose to reset a level, in which case the original configuration is restored (of Ninja-se and the keys and locked walls).

A move counter keeps track of Ninja-se's progress. With each move, the counter is increased. When the player resets a level, the move counter is reset to zero. The player's move count is always visible.

The keys (and corresponding locked walls) can be colored RED (#ff0000), YELLOW (#ffff00), GREEN (#00ff00), BLUE (#0000ff), CYAN (#00ffff), MAGENTA (#ff00ff). There may be multiple keys with the same color.

When all locked walls have been cleared, a congratulatory message is shown to the player.

The third and final configuration appears below which can be solved in 16 moves.



As an aside: Can you find a 2x5 arrangement that requires more moves? No points, just bragging rights...

Template to follow

Use Case: [Verb Noun]

- Participating Actor: Initiated by Actor1
- Entry Condition
  - List here
- Exit Criteria
  - Updated State of system (NOT like a functional call return)
- Flow of Events
  1.Initiating Actor performs some action
  2.System responds in kind [even step]
  3.Participating Actor performs some action
  4.System responds in kind [even step]

Use Cases to Write

1. Choose configuration
2. Move Ninja-Se
3. Pick up Key
4. Unlock Door
5. Reset Puzzle
6. ~~Solve Puzzle~~

IP.1 Due Wednesday November 2th at 1:00 PM (5% of total grade)

The first deliverable is due in the second week of the course. This deliverable will contain:

- A set of use cases, formatted and structured using the examples presented in class
- A mock-up interface (also called a storyboard) that graphically shows what the interface will look like. Identify each actionable element on the mockup and explain (briefly) how player interacts to perform the necessary tasks.
- [I moved this text to IP.2 which is where it should have been all along] ~~In representing the board:~~
    - ~~You can simplify your rendering by showing the location of Ninja-Se by drawing the cell in Purple (#a349a4).~~
    - ~~When Ninja-se is holding a key you need to visually show the color of the key he holds, which you can do by drawing a small colored square on top of Nina-se~~
    - ~~If Ninja-se is in a cell and he has dropped a key when picking up a new one, you do not have to visually show the key that was dropped because it is obscured by Ninja-se (naturally, should Ninja-se leave that cell, it would then be visible behind him)~~. [End Moving Txt]

Please submit a single file (such as a .pdf or a MS-word .docx) which contains the formatted use cases as I've described. It would be best if the name of the file included your own name (to avoid clashes).

What comes next?

Assignment IP.2 is due Monday November 14th at 4:59 PM. Be aware of the deadlines!

Updates

- **Will be timestamped and posted here with changes above in red.**
- **Note: Use case "Solve Puzzle" has been stricken from the record and is no longer part of this assignment!**