Randomness and bounds checking in a simple snake game: Java Snakes

Java Snakes is based on a classic early computer game:

Navigate your snake, avoiding walls, your tail and the tail of your opponent. Try to box in your opponent, causing them to crash. Each player has 3 lives.

Powerups may appear to be consumed by a player:

\*Flower powerups will make the recipient grow. The number in the flower denotes the size that the snake will grow. In some game modes, the game will speed up after every 8 powerups eaten.

\*Blue flowers will cause the snake to separate from its tail, leaving it as a wall in the map. This way, the map changes dynamically and gets more complex as the round goes on.

\*Red flowers will give the snake a larger head, and with it the ability to run through one wall or snake tail. When striking a snakes tail with the Red powerup, the tail will separate at the point of impact and be left as a wall.

As it stands, the powerups always spawn in the center of the screen, and the program will crash if a snake tries to warp from one edge of the board to another. Not a very fun game. You, the student, will be responsible for completing some of the algorithms to get the game in good working order. **Test your code by compiling and running JavaSnakesDriver.java.** Here are your tasks:

FIX 1a, Fix 1b) in *SnakePanel.java*, complete the method void spawnPellet():

This has the responsibility of spawning a growth pellet and/or a powerup into the world. It currently places them in the center. Add code to make the pellets and powerups spawn in a valid random location on the map. In the event that it picks a location that has a wall covering it, the program needs to pick another spot.

Fix 2) in *Player.java*, complete the method void *move*():

This has the responsibility of looking at the direction the player picked (up, down, left or right) and updating their position, denoted by row and col. Currently, if a player tries to warp to the other side of the board, the program will crash by throwing an ArrayIndexOutOfBoundsException.

This is bad. Insert code to provide a wrap-around effect. If the player tries to move up from the top edge of the board, teleport them to the bottom. Likewise, if a player tries to move left from the left side of the board, teleport them to the right side. Do the same for all board edges.

The first row is denoted by 0 and the first col is also 0.

The last row is denoted by (numRows-1) and the last col is (numCols-1).