Crowd dynamics simulation - a multi-agent system based on CA

Preparation:

- Use the project code provided in the course resources as a basis: **Pedestrians.zip**
- Import and run the project.
- Customize the code to meet the requirements of the assignment.

Cell types in CA:

0 - floor

1 - wall

2 - exit

3 - pedestrian

Neighborhood:

In class **Board** in method **initialize()** initialize neighbors for every cells. Prepare two versions using:

- Moore neighborhood,
- von Neuman neighborhood.

Do not initialize neighbors for the border cells.

Static potential field (Score: 2 points)

In class **Board** in method **calculateField()**:

- Create list of **Points** for which static field should be recalculated. Please note that initially the value of **staticField** is set to **SFMAX** (100000).

  ```
  ArrayList<Point> toCheck = new ArrayList<Point>();
  ```

  - For each cell of type 2 (exit), set its **staticField** to 0. Each neighbor of such cell should be added to **toCheck** list.
  - Until list **toCheck** is empty:

    1. Verify if first element on **toCheck** list changes its **staticField** (use method **calcStaticField()** for this cell)

2. If it is true, add all neighbors of this cell to **toCheck** list.
3. Remove first element from the list.

In class **Point**:

- Implement method **calcStaticField()**. If this cell **staticField** is larger than smallest value of neighbours **staticField + 1**, set cell **staticField** to this value. Return true if you change the value of **staticField**, otherwise return false.

Run your application. Set some exit. Push button „Calc Field". Analyse the influence of used neighborhood to the shape of static floor field.

Crowd dynamics - naive implementation (Score: 2 points)

In class Point in method move():

- Check if there is a pedestrian in given cell:

```
isPedestrian == true
```

- If so, move the pedestrian to the not occupied, neighbor cell with smallest static floor field.

Run simulation. Observe what artifacts appears. Try to find reasons of this errors.

Crowd dynamics - improvements (Score: 2 points)

Main reasons behind errors in previous point are:

- Agents that reach exit should be removed from the simulation. If pedestrian enters cell type 2 (exit) do not set its variable **isPedestrian** on true.
- No synchronization of cells. You may notice that moving agents sometimes get lost or go in the wrong direction.
  In order to fix this issue, create in class **Point** variable

  ```
  boolean blocked = false;
  ```

  If cell is blocked, agent on this cell can't make a move. Cell is blocked if some pedestrian enters it. Remember to unblock all cells at the beginning of each iteration.

  In addition, to eliminate erroneous behavior, the possibility of going outside the simulation area should be blocked either by setting the appropriate cells to type 1 (wall) and prohibiting entry into such an area, or by setting a sufficiently large value of staticField (SFMAX) on the boundaries of the area.

After completing the implementation and verifying correct operation for both variants of the environment, think about further possible improvements that could be applied:

- Improve calculation method for static floor field.
- Add repulsion force between pedestrians and walls (add it by modification of static floor field close to the walls).
- Random order of pedestrians movement.
- ... your idea.