//public class SimpleMaze3dGenerator extends CommonMaze3dGenerator {

//

// @Override

// public Maze3d generate(int z, int y, int x) {

// Maze3d maze = new Maze3d(z, y, x);

// Random rand = new Random();

//

// // Choose a random start position and turn in to a free cell

// Position pos = chooseRandomPosition(z, y, x);

// maze.setStartPosition(pos);

// maze.setCellValue(pos, Maze3d.FREE);

//

// int counter = z + y + x;

// while (counter >= 0) {

// // Get all neighbors that are walls

// ArrayList<Position> allNeighbors = getNeighborsWithoutWalls(pos, z, y, x);

// ArrayList<Position> neighbors = maze.getNeighborsByValue(allNeighbors, Maze3d.WALL);

//

// // If there are walls neighbors, choose random one and turn it to

// // free cell

// if (neighbors.size() > 0) {

// pos = neighbors.get(rand.nextInt(neighbors.size()));

// maze.setCellValue(pos, Maze3d.FREE);

// counter--;

// } else

// break;

// }

// // Set the last chosen position to be the goal position

// maze.setGoalPosition(pos);

// return maze;

// }

//\*/

//

// /\*\*

// \* <h1>getNeighborsWithoutWalls</h1> Create a list of position received

// \* neighbors.

// \* <p>

// \*

// \* @param p

// \* Position in maze. A list of it's neighbors is return

// \* @param z

// \* Total floors in the 3D maze

// \* @param y

// \* Total rows is the 3D maze

// \* @param x

// \* Total columns in the 3D maze

// \* @return A list of position received (p) neighbors

// \*/

//

// private ArrayList<Position> getNeighborsWithoutWalls(Position p, int z, int y, int x) {

// ArrayList<Position> neighbors = new ArrayList<Position>();

// if (p.z > 0)

// neighbors.add(new Position(p.z - 1, p.y, p.x));

// if (p.z < (z - 1))

// neighbors.add(new Position(p.z + 1, p.y, p.x));

// if (p.y > 0)

// neighbors.add(new Position(p.z, p.y - 1, p.x));

// if (p.y < (y - 1))

// neighbors.add(new Position(p.z, p.y + 1, p.x));

// if (p.x > 0)

// neighbors.add(new Position(p.z, p.y, p.x - 1));

// if (p.x < (x - 1))

// neighbors.add(new Position(p.z, p.y, p.x + 1));

// return neighbors;

// }

//}