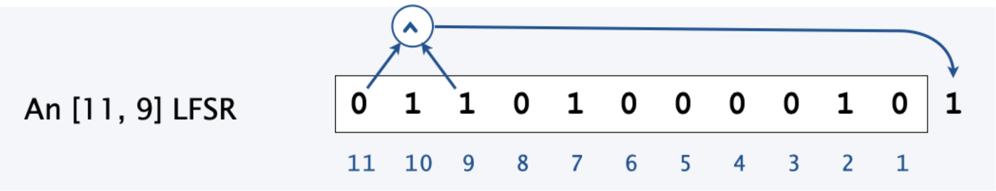
CSC1003 Practice Outline Oct 17-21

Programming exercises about arrays.

1. Use array(s) to simulate the Linear Feedback Shift Register (LFSR) with tap positions at 11 and 9, and the contents in the below figure as the initial fill.

   Hint: you can find the reference code in the lecture note "CS.0.Prologue".



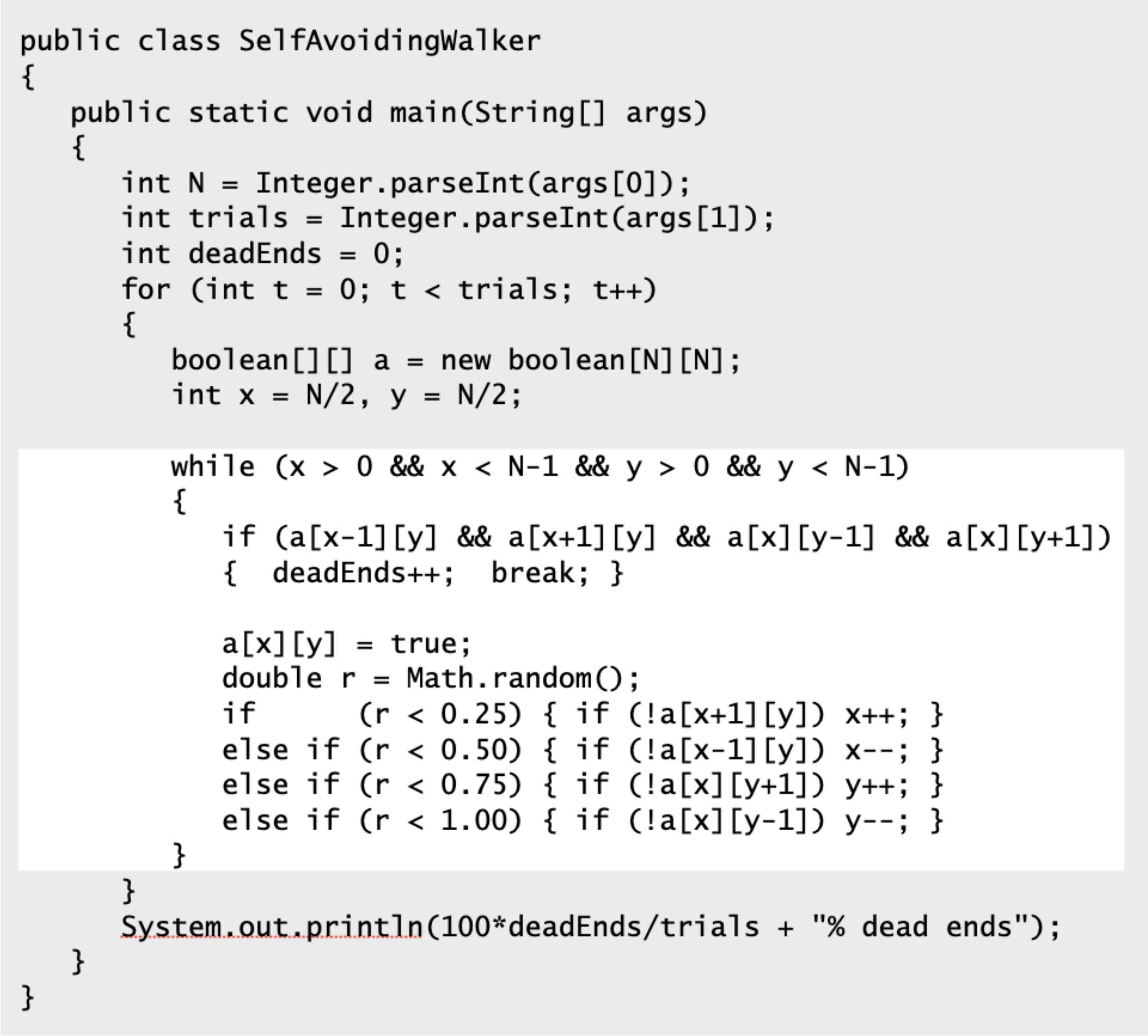
2. Read n numbers from the command arguments, and output the in reverse order.

   For example, if you give 5 integers 1 2 3 4 5 as the input, the output should be 5 4 3 2 1.

   You can also use Scanner to get the input from standard input stream, if you want.

3. Modify SelfAvoidingWalker to calculate the average length of the paths that reaches a dead end. For example, suppose we have 10 trials, if 3 paths reach the dead end with lengths of 10, 20, and 15, then the average is 15.

   You can also compute the average length to escape from the lattice.



```
public class SelfAvoidingWalker
{
   public static void main(String[] args)
   {
      int N = Integer.parseInt(args[0]);
      int trials = Integer.parseInt(args[1]);
      int deadEnds = 0;
      for (int t = 0; t < trials; t++)
      {
         boolean[][] a = new boolean[N][N];
         int x = N/2, y = N/2;

         while (x > 0 && x < N-1 && y > 0 && y < N-1)
         {
            if (a[x-1][y] && a[x+1][y] && a[x][y-1] && a[x][y+1])
            {  deadEnds++;  break; }

            a[x][y] = true;
            double r = Math.random();
            if      (r < 0.25) { if (!a[x+1][y]) x++; }
            else if (r < 0.50) { if (!a[x-1][y]) x--; }
            else if (r < 0.75) { if (!a[x][y+1]) y++; }
            else if (r < 1.00) { if (!a[x][y-1]) y--; }
         }
      }
      System.out.println(100*deadEnds/trials + "% dead ends");
   }
}
```

4. a) Suppose that a random walker, starting in the center of an *n*-by-*n* grid, move one step at a time, choosing to go left, right, up, or down with equal probability at each step. Write a program to simulate the process and compute the average number of steps taken before all cells are touched by repeat the trials for *T* times. *n* and *T* can be given by command-line arguments (or standard input).

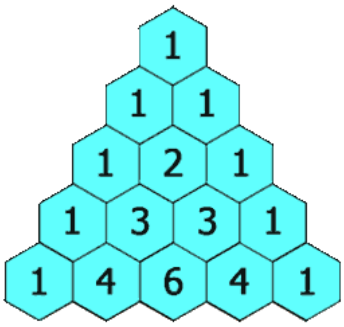
   Hint: you can use a 2-dimensional boolean array to record whether the corresponding cells are touched or not. Refer SelfAvoidingWalker in lecture note.

   b) [Optional] If we have *n* random walkers, starting from the center, their moves are independent from each other, what is the average number of steps taken before all cells are touched?

   Hint: apart from the 2-dimentional boolean array used in a), you can use two more 1-

dimensional int arrays to track the current position (indexes in the lattice) of each random walker.

5. [Optional] Given an integer N, print the first N rows of **Pascal's triangle**. In **Pascal's triangle**, each number is the sum of the two numbers directly above it as shown.



If n is 5, the the first 5 rows of your program output looks like the following:

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
```

Hint: you can use a 2-dimensional array to implement this program.