Alexander Zubkov 346142375
Alexey Rozman 342852639
Anastasia Iksar 345181192
Dima Pekar 345250005
Nikita Orlov 320547003

## Program code

```
public class Main {

	public static void main(String[] args) {

		//double p = 0.5;
		double[] p = {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99};
		int m1 = 1000;
		int m2 = 10000;
		int[] terminals = {16, 12, 0};

		System.out.println("p\tM=1000\tM=10000");
		for (double pi : p) {
			System.out.print(pi + "\t");
			System.out.printf("%.4f\t",
Tools.calculateNetworkReliability(terminals, m1, pi));
			System.out.printf("%.4f\t",
Tools.calculateNetworkReliability(terminals, m2, pi));
			System.out.println();
		}



	}

}


public class Tools {

	/*
	 *    01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
	 * 01  0  1  0  0  1  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0
	 * 02  1  0  1  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0
	 * 03  0  1  0  1  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0
	 * 04  0  0  1  0  1  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
	 * 05  1  0  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
	 * 06  0  0  0  0  1  0  1  0  0  0  0  0  0  0  1  0  0  0  0  0
	 * 07  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  0  1  0  0  0
	 * 08  1  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0
	 * 09  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  1  0  0
	 * 10  0  1  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0  0  0
	 * 11  0  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  1  0
	 * 12  0  0  1  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  0
	 * 13  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  1
```

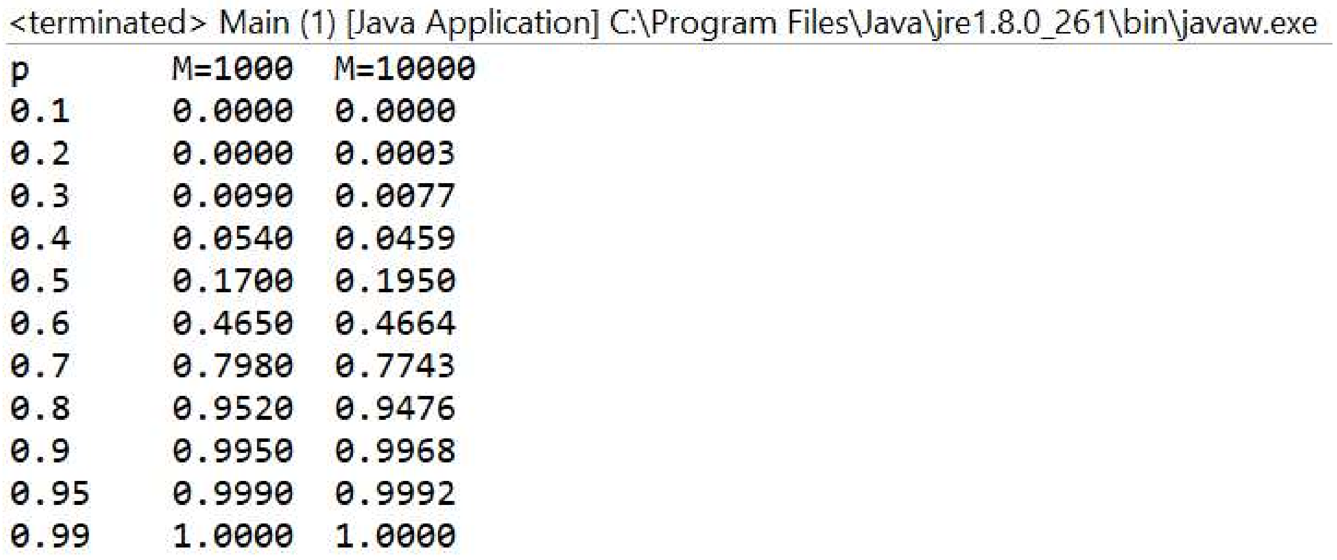
```
         * 14  0  0  0  1  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0
         * 15  0  0  0  0  0  1  0  0  0  0  0  0  0  1  0  1  0  0  0  0
         * 16  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  1
         * 17  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  1  0  1  0  0
         * 18  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  1  0  1  0
         * 19  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  1  0  1
         * 20  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  1  0  0  1  0
         */

        static private double[] p = {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95,
0.99};
        static private int[][] connections = {
                        /*00*/{0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*01*/{1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*02*/{0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*03*/{0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0},
                        /*04*/{1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*05*/{0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
                        /*06*/{0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0},
                        /*07*/{1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*08*/{0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0},
                        /*09*/{0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0},
                        /*10*/{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0},
                        /*11*/{0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0},
                        /*12*/{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1},
                        /*13*/{0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0},
                        /*14*/{0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0},
                        /*15*/{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1},
                        /*16*/{0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0},
                        /*17*/{0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0},
                        /*18*/{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1},
                        /*19*/{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0}
        };
        static private int[][] stateVector = new int[connections.length]
[connections[0].length];

        static public double calculateNetworkReliability(int[] terminals, int m, double p) {

                double r = 0;

                for (int i = 0; i < m; i++) {
                        calculateNetworkStateVector(p);
                        if (checkTerminalsConnection(terminals))
                                r++;
                }

                //System.out.println(r); //print count of "UP" results (for tests)

                return r / m;
        }

        static private void calculateNetworkStateVector(double p) {

                if (connections == null || connections.length <= 0 || connections[0].length !
= connections.length ) {
                        System.out.println("Incorrect connections array");
                        return;
                }
```

```
		int length = connections.length;
		//int[][] stateVector = new int[length][length];

		for (int i = 0; i < length; i++) {
			for (int j = i; j < length; j++) {
				if (connections[i][j] == 1)
					stateVector[i][j] = Math.random() <= p ? 1 : 0;
				else
					stateVector[i][j] = 0;
			}
		}

		for (int i = 0; i < length; i++) {
			for (int j = 0; j < i; j++) {
				stateVector[i][j] = stateVector[j][i];
			}
		}

		//print transition matrix (for tests)
		/*
		int r = 0;
		for (int i = 0; i < length; i++) {
			System.out.print((i < 10 ? "0" + i : i) + "|  ");
			for (int j = 0; j < length; j++) {
				if (stateVector[i][j] == 1 && i < j)
					r++;
				System.out.print(stateVector[i][j] + "  ");
			}
			System.out.println();
		}
		*/
		//System.out.println(r); //print count of "UP" results (for tests)
	}

	static private boolean checkRoute(int x, int y) {
		if (x > y) {
			int tmp = x;
			x = y;
			y = tmp;
		}

		return checkRoute(x, y, " " + x + " ");
	}

	static private boolean checkRoute(int x, int y, String route) {

		if (x == y) {
			//System.out.println(route.substring(1)); //print route if exists (for
tests)
			return true;
		}

		for (int i = 0; i < stateVector.length; i++) {
			if (stateVector[x][i] == 1
					&& !route.contains(" " + i + " ")
					&& checkRoute(i, y, route + i + " "))
				return true;
```

```
                }

                return false;
        }

        static private boolean checkTerminalsConnection(int[] terminals) {
                for (int i = 0; i < terminals.length; i++)
                        for (int j = i + 1; j < terminals.length; j++)
                                if (!checkRoute(terminals[i], terminals[j]))
                                        return false;

                //System.out.println("true"); //true - network is UP (for tests)
                return true;
        }

}
```

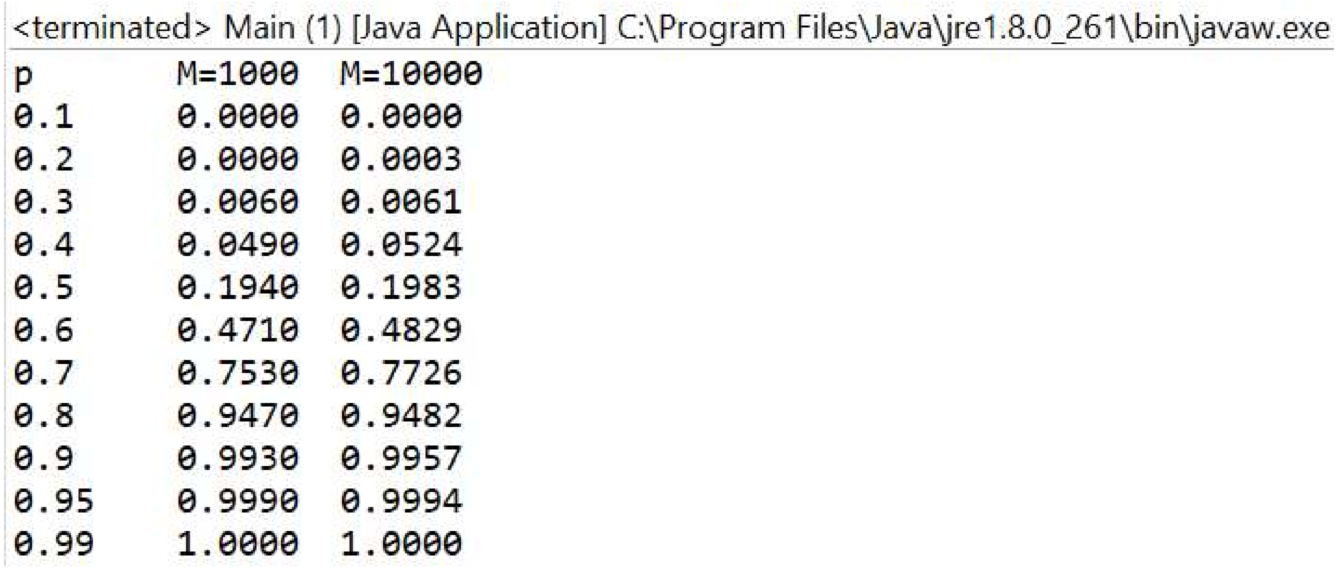
## Program result

Run 1:



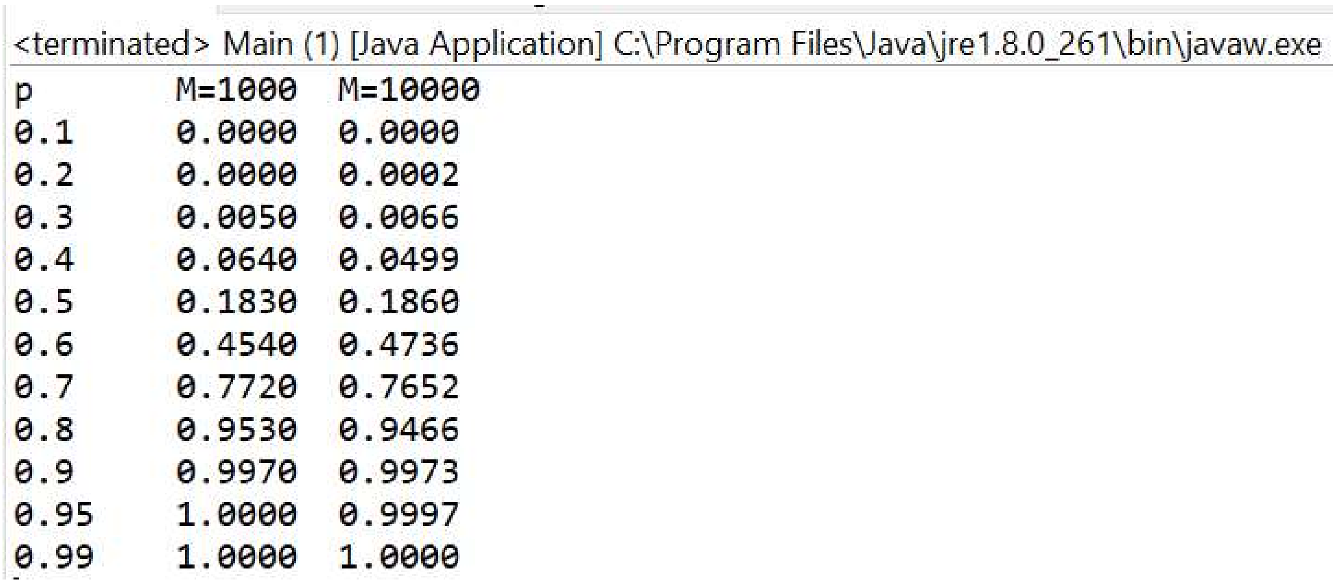
```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe
p       M=1000  M=10000
0.1     0.0000  0.0000
0.2     0.0000  0.0003
0.3     0.0090  0.0077
0.4     0.0540  0.0459
0.5     0.1700  0.1950
0.6     0.4650  0.4664
0.7     0.7980  0.7743
0.8     0.9520  0.9476
0.9     0.9950  0.9968
0.95    0.9990  0.9992
0.99    1.0000  1.0000
```

Run 2:



```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe
p         M=1000   M=10000
0.1       0.0000   0.0000
0.2       0.0000   0.0003
0.3       0.0060   0.0061
0.4       0.0490   0.0524
0.5       0.1940   0.1983
0.6       0.4710   0.4829
0.7       0.7530   0.7726
0.8       0.9470   0.9482
0.9       0.9930   0.9957
0.95      0.9990   0.9994
0.99      1.0000   1.0000
```

Run 3:



```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe
p         M=1000   M=10000
0.1       0.0000   0.0000
0.2       0.0000   0.0002
0.3       0.0050   0.0066
0.4       0.0640   0.0499
0.5       0.1830   0.1860
0.6       0.4540   0.4736
0.7       0.7720   0.7652
0.8       0.9530   0.9466
0.9       0.9970   0.9973
0.95      1.0000   0.9997
0.99      1.0000   1.0000
```