CS 401 Algorithms **April 26, 2019**
Assignment2
Due Date: May 10, 2019

Individual/Group Assignment. You can work in a team (at most 2 people) or alone. Teams: please let me know the members of the team by April 30th.

Part I: The Ability of Soil to Hold Water

Soil scientists characterize and classify soils into different groups. Each group of soil has differing ability to hold water depending on the particles inside. For the sake of simplicity, you are given the internal structure of soil as *n*-by-*n* grid of *cells*. Each cell is either *1* or *0*.

- 1 means, the cell allows water to drain.
- 0 means that cell will hold the water.

You are asked to write a Java program. The program shall

- Read n-by-n grid of cells from a text file
- Determine if the soil allow the water to drain or hold the water
    - Prints "Allow water to drain" or "Don't allow water to drain" as a console output.

You are asked to use the union-find algorithm to solve whether a soil example allows water to drain.

Sample Input File-1:

```
1   0   1   0   1
1   1   0   1   0
0   1   1   0   1
1   0   1   0   1
1   0   1   1   1
```

Sample Output-1:

```
Allows water to drain
```

Sample Input File-2:

```
1   0   0   1   1
0   1   1   1   0
1   0   0   0   1
1   0   0   0   1
1   1   0   1   1
```

Sample Output-2:

```
Don't allow water to drain
```

Sample Input File-3:

```
1   0   0   1   1
0   1   1   1   0
1   0   1   1   1
0   0   0   1   1
1   1   0   0   0
```

Sample Output-3:

```
Don't allow water to drain
```

Sample Input File-4:

```
1 0 0 1 1 1 1 0 0 0
0 1 1 1 0 0 0 1 1 0
1 0 0 1 1 0 0 1 0 0
1 0 0 0 1 1 1 0 0 0
1 1 0 1 1 1 1 0 0 0
1 0 1 1 0 1 1 1 1 0
0 0 0 0 1 1 0 0 0 0
1 0 1 1 1 1 1 0 0 0
0 1 0 1 1 0 1 0 1 0
1 1 0 1 0 1 1 0 0 0
```

Sample Output-4:

```
Allows water to drain
```

EXTRA 10-Point: If you implement and use Weighted Quick Union with Path Compression in your solution, you'll get extra 10 points.

Part II: Sorting Algorithm GUI Application

Create a **Java GUI** application which meets the following user requirements:

- User enters a data size, N (Assume N<500000)
- User selects the type of file to be generated: sorted or random
- If user selects *sorted* option, the application generates N integers and save the numbers in sorted order in a file named as *sorted500000.txt*
- If user selects *random* option, the application shall generate N random integers and save them in a random order into a file named random500000.txt
- User selects a sorting algorithm from 7 sorting algorithms including selection sort, insertion sort, shell sort, bubble sort, merge sort, quick sort and heap sort.
- User presses "SORT" button
- The application sorts the generated data in the file with the selected sorting algorithm
- Then, the application displays the running time of the sorting operation on the graphical user interface.

HOW TO SUBMIT

You are supposed to submit your work as a single zip file via CANVAS. Zip file will include the following two archive files for each part:

- Part1.zip
- Part2.zip

Please use the following file format while naming the zip file:
LastNameFirstnameX_Y.zip where LastNameFirstname is your last name with the first letter in capital, followed by your first name with the first letter in capital; the X is the course code; the Y is the assignment #. (ex: SerceFatmaCS401_2.zip)