INF 551 – Spring 2017

Homework #1

Due: February 10, Friday
100 points

Task: In this problem, you are asked to implement in Python2.7 the following disk scheduling algorithms: *SSTF*, *LOOK*, and *C-SCAN*, as described in class. Recall that a disk scheduler takes a queue of requests (may contain **identical** ones), each represented by the location of its track, and produces an ordering (i.e., schedule) of requests. We assume that the platter has 200 tracks, numbered from 0 to 199.

You may assume that the disk is currently idle. Thus, in *LOOK* algorithm, the disk head will first move in the direction to the track closest to head, among all requests. Note that the disk head moves only in one direction in *C-SCAN*: from innermost track (track #0) to outermost (track #199).

Input: The program takes as input a text file, e.g., *queue.txt*, formatted as follows.

- First line is the number of the track where the disk head is currently located;
- Second line is the list of requests, separated by commas and NO spaces in between.

For instance, here is the content of *queue.txt* for the example shown in class.

```
50
120,30,60,135,80,20,95,160,70,5
```

Output: For each algorithm, output the ordering of requests and the cost, i.e., the total number of tracks to be travelled. Note that the cost of *C-SCAN* includes the tracks jumped over by the head (see examples in the lecture). Your algorithm should output two lines. First line shows the schedule; the second line is the cost. For example, *python sstf.py queue.txt* will output:

```
60,70,80,95,120,135,160,30,20,5
265
```

Execution: Use the command line below to execute your scripts.

python <your_script_name>.py <sample>.txt

Submissions: Name your 3 scripts as below and submit to blackboard by the due time. **DO NOT** make them into folder or zip file.

- <FirstName>_<LastName>_sstf.py
- <FirstName>_<LastName>_look.py
- <FirstName>_<LastName>_c-scan.py

Special Case: For *SSTF* if the head is the same distance away from two requests, follow this strategy:
(1) If it is the first move (**beware** of the situation where initial head location is the same as one or several request(s)), move the head toward the nearest end, e.g., if it is positioned at 99, it should move toward the direction where 0 is (also applies to *LOOK*);
(2) Else, move in the same direction as in the previous step.