Birla Institute of Technology & Science, Pilani

Work-Integrated Learning Programmes Division
MTech in Data Science & Engineering
S1_2023-2024, DSECTZG519- Data Structures & Algorithms Design

Assignment 1 – PS1 – [Construction of a BITS Pilani Tower] - [Weightage 12%]

Read through this entire document very carefully before you start!

1. Problem Statement

BITS Pilani is going to celebrate its 60th anniversary in a couple of years. To mark the magical days at BITS Pilani, students are planning to construct a Tower. Your task is to construct this tower in N days by following these conditions:

- Every day you are provided with one disk of distinct size.
- The disk with larger sizes should be placed at the bottom of the tower.
- The disk with smaller sizes should be placed at the top of the tower.

The order in which tower must be constructed is as follows:

- You cannot put a new disk on the top of the tower until all the larger disks that are given to you get placed.

Requirements:

1. Implement the above problem statement using Python 3.7
2. Read the input from a file(**inputPS1.txt**), Print N lines denoting the disk sizes that can be put on the tower on the ith day. Use Queues to solve the statement.
3. You will output your answers to a file (**outputPS1.txt**) for each line.
4. Perform an analysis for the features above and give the running time in terms of input size: n.

Sample Input:

First line: N denoting the total number of disks that are given to you in the N subsequent days. Second line: N integers in which the i th integers denotes the size of the disks that is given to you on the i th day
Note: All the disk sizes are distinct integers in the range of 1 to N.

Input will be taken from the file(**inputPS1.txt**)

```
5
4 5 1 2 3
```

Note that the input/output data shown here is only for understanding and testing, the actual file used for evaluation will be different.

Sample Output:

Print N lines. In the i th line, print the size of disks that can be placed on the top of the tower in descending order of the disk sizes.

If on the i th day no disks can be placed, then leave that line empty.

Display the output in **outputPS1.txt**.

```
1 >
2 > 5 4
3>
4>
5> 3 2 1
```

Note that the input/output data shown here is only for understanding and testing, the actual file used for evaluation will be different with all the test cases.

Explanation

- On the first day, the disk of size **4** is given. But you cannot put the disk on the bottom of the tower as a disk of size **5** is still remaining.
- On the second day, the disk of size **5** will be given so now disk of sizes **5** and **4** can be placed on the tower.
- On the third and fourth day, disks cannot be placed on the tower as the disk of **3** needs to be given yet. Therefore, these lines are empty.
- On the fifth day, all the disks of sizes **3**, **2**, and **1** can be placed on the top of the tower.

2. Deliverables:

1. PDF document **designPS1_<group id>.pdf** detailing your design approach and time complexity of the algorithm and alternate solutions.

2. **[Group id] _Contribution.xlsx** mentioning the contribution of each student in terms of percentage of work done. Columns must be "Student Registration Number", "Name", "Percentage of contribution out of 100%". If a student did not contribute at all, it will be 0%, if all contributed then 100% for all.
3. **inputPS1.txt** file used for testing
4. **outputPS1.txt** file generated while testing
5. **.py file** containing the python code. Create a single *.py file for code. Do not fragment your code into multiple files.
6. **Zip all of the above files including the design document and contribution file in a folder with the name: [Group id]_A1_PS1.zip** and submit the zipped file in canvas.
7. **Group Id** should be given as **Gxx** where xx is your group number. For example, if your group is 26, then you will enter G26 as your group id.

3. Instructions

1. It is compulsory to make use of the data structure(s) / algorithms mentioned in the problem statement.

2. Ensure that all data structure insert and delete operations throw appropriate messages when their capacity is empty or full. Also ensure basic error handling is implemented.

3. For the purposes of testing, you may implement some functions to print the data structures or other test data. But all such functions must be commented before submission.

4. Ensure that the input, prompt and output file guidelines are adhered to. Deviations from the mentioned formats will not be entertained.

5. The input, prompt and output samples shown here are only a representation of the syntax to be used. Actual files used to evaluate the submissions will be different. Hence, do not hard code any values into the code.

6. Run time analysis is to be provided in asymptotic notations and not timestamp based runtimes in sec or milliseconds.

Instructions for use of Python:

1. Implement the above problem statement using Python 3.7.

2. Use only native data types like lists and tuples in Python, do not use dictionaries / built-in functions provided in Python. Use of external libraries like graph, numpy, pandas library etc. is not allowed. The purpose of the assignment is for you to learn how these data structures are constructed and how they work internally.

3. Create a single *.py file for code. Do not fragment your code into multiple files.

4. Do not submit a Jupyter Notebook (no *.ipynb). These submissions will not be evaluated.

5. Read the input file and create the output file in the root folder itself along with your .py file. Do not create separate folders for input and output files.

4. Deadline

1. The strict deadline for submission of the assignment is **3rd Dec 2023 (Sunday) 11:55 PM IST.**

2. The deadline has been set considering extra days from the regular duration in order to accommodate any challenges you might face. No further extensions will be entertained.

3. Late submissions or offline submissions will **NOT** be evaluated.

5. How to submit

1. This is a group assignment.

2. Each group has to make one submission (**only one, no resubmission**) of solutions.

3. Each group should zip all the deliverables in one zip file and name the zipped file as mentioned above.

4. Assignments should be submitted via Canvas > Assignment section. Assignments submitted via other means like email, inbox etc. will not be graded. No discussions with regard to this will be entertained.

6. Evaluation

1. The assignment carries 12 Marks.

2. Grading will depend on

   a. Fully executable code with all functionality working as expected

   b. Well-structured and commented code

   c. Accuracy of the run time analysis and design document.

3. Every bug in the functionality will have negative marking. We will test any possible case.

4. Marks will be deducted if your program fails to read the input file used for evaluation due to change / deviation from the required syntax.

5. Use of only native data types and avoiding libraries like numpy, graph and pandas will also be a rubric.

6. We encourage students to take the upcoming assignments and examinations seriously and submit only original work. Please note that plagiarism in assignments will be taken seriously. All groups that are booked under plagiarism will be given 0 marks and no further discussion will be entertained. Please refer to the detailed policy in the files section of canvas.

7. Source code files which contain compilation errors will get at most 25% of the value of that question.

7. Readings

Text book: Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, Wiley (Students Edition). **Chapters:** 2

************ All the Best ************