Work Integrated Learning Programmes Division
M.Tech (Data Science and Engineering)

(S2-20_DSECLZG519)
(Data Structures and Algorithms Design)
Academic Year 2020-2021

Assignment 1 – PS5 - [Company Subsidiaries] - [Weightage 12%]

1. Problem Statement

You are working for a giant conglomerate. One way of increasing their presence and utilize free cash is by investing/buying and selling companies. Sometimes an acquired company could invest and buy another smaller company and sometimes could also sell a company.

For example, consider a conglomerate Common Electric (CE) that has acquired 3 companies:

1. Aviation sector (for making planes and engines)
2. Power sector (build and manages power plants)
3. Healthcare sector (build and provides healthcare related equipment)

After sometime the aviation sector acquired another company called 'Additive' that has the capability of building 3D printers and use them to build complex parts required. Similarly, the Power sector saw bought appliance manufacturers, solar power and wind power related companies. Power sector also bought another company that helped with dominating its presence in a particular geographical location. Although, later it turns out to be their biggest mistake.

You are required to keep a record of companies bought and sold such that each node represents a company and the immediate child nodes are the companies bought by the conglomerate division. Implement following functions:

a. **detail**: Display the details of the mentioned company. Who acquired the company and what other companies were acquired by the mentioned company
b. **acquire**: Adds a new acquired company to the records.
c. **release**: Release or sell off the mentioned company

You can assume that there cannot be 2 companies with the same name.

Requirements

1. Implement the above problem statement as a General Tree using Linked List and Python 3.7.
2. Read the input from a file **inputPS5.txt.**
3. You will output your answers to a file **outputPS5.txt**
4. Perform an analysis for the features above and give the running time in terms of input size: n.

```
def detail(company_name)
    """this function prints the parent and immediate children of company
    """

def acquire(parent_company, acquired_company)
    """Inserts the acquired_company as a new child node to the parent_company
    """

def release(released_company)
    """removes the node mentioned in the released_company.
    """
```

Sample Input and Output

You will be given N (a number) representing the number of operations to be performed.

Followed by N number of operations. Each line will contain one operation.

Sample Input file	Sample Output file
Company: ce No of operations: 6 DETAIL ce ACQUIRED:aviation BY:ce ACQUIRED:power BY:ce ACQUIRED:healthcare BY:ce DETAIL ce ACQUIRED:additive BY:aviation ACQUIRED:additive BY:aviation DETAIL aviation RELEASE additive RELEASE additive ACQUIRED:additive BY:aviation ACQUIRED:wind-energy BY:power ACQUIRED:solar-energy BY:power ACQUIRED:appliances BY:power ACQUIRED:ct-manufacturer BY:healthcare ACQUIRED:lifescience BY:healthcare ACQUIRED:pharma BY:healthcare DETAIL ce DETAIL aviation DETAIL power DETAIL lifescience	DETAIL: ce Acquired companies: none No of companies acquired: 0 ACQUIRED SUCCESS: ce Successfully acquired aviation ACQUIRED SUCCESS: ce Successfully acquired power ACQUIRED SUCCESS: ce Successfully acquired healthcare DETAIL: ce Acquired companies: aviation, power, healthcare No of companies acquired: 3 ACQUIRED SUCCESS:additive BY:aviation ACQUIRED FAILED:additive BY:aviation DETAIL: aviation Acquired companies: additive No of companies acquired: 1 RELEASED SUCCESS: released additive successfully. RELEASED FAILED: released additive failed. ACQUIRED SUCCESS: aviation Successfully acquired additive ACQUIRED SUCCESS: power Successfully acquired wind-energy ACQUIRED SUCCESS: power Successfully acquired solar-energy ACQUIRED SUCCESS: healthcare Successfully acquired aviation ACQUIRED SUCCESS: healthcare Successfully acquired ct-manufacturer ACQUIRED SUCCESS: healthcare Successfully acquired pharma DETAIL: ce Acquired companies: aviation, power, healthcare No of companies acquired: 3 DETAIL: aviation Acquired companies: additive No of companies acquired: 1 DETAIL: power Acquired companies:wind-energy, solar-energy, appliances No of companies acquired: 3 DETAIL: lifescience Acquired companies: none No of companies acquired: 0

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

2. Deliverables

1. Word document **designPS5_<group id>.docx** detailing your design and time complexity of the algorithm.
2. **[Group id]_Contribution.xlsx** mentioning the contribution of each student in terms of percentage of work done. Download the Contribution.xlsx template from the link shared in the Assignment Announcement.
3. **inputPS5.txt** file used for testing
4. **outputPS5.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

Zip all of the above files including the design document and contribution file in a folder with the name:

[Group id]_A1_PS5_CompanySubsidiaries.zip and submit the zipped file.

Group Id should be given as **Gxxx** where xxx is your group number. For example, if your group is 26, then you will enter G026 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. Make sure that your read, understand, and follow all the instructions
5. Ensure that the input, prompt and output file guidelines are adhered to. Deviations from the mentioned formats will not be entertained.
6. 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.
7. Run time analysis is to be provided in asymptotic notations and not timestamp based runtimes in sec or milliseconds.
8. Please note that the design document must include
    a. The data structure model you chose with justifications
    b. Details of each operations with the time complexity and reasons why the chosen operations are efficient for the given representation
    c. One alternate way of modelling the problem with the cost implications.

9. Writing good technical report and well document code is an art. Your report cannot exceed 4 pages. Your code must be modular and quite well documented.

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 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 **Monday, 21th Jun, 2021.**
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 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. Assignment submitted via other means like email etc. will not be graded.

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.

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 get additional marks.
6. **Plagiarism will not be tolerated. Copy / Paste's from web resources / or your friends' submission will attract severe penalty to the extent of awarding negative 10 percent. We will not measure the extent of such blatant copy pastes and details of who copied from whom and such details while awarding the penalties. It's the responsibility of the team to solve and protect your original work.**
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:** 2.3