BITS Pilani
Pilani | Dubai | Goa | Hyderabad

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

(Data Structures and Algorithms Design)

Academic Year 2022-2023

Assignment – PS11-[Weightage 25%]

1. Problem Statement –Question 1

In a company named "ABC", the employees are identified by their employee ID and name. The employee details are stored in a binary search tree data structure format. The employee details contain Employee name, employee ID and designation. The details are inserted into the binary search tree data type based on the employee ID (employee ID is an integer data type). The employee Id is assigned based on the joining date of the employee. For example, the employee who joined latest will have the highest employee ID.

The following operations are expected to perform

1. Employee name and their details are stored in a file "abc_emp_dataPS11.txt". Program should read the data from the file and create a BST with each node having the following details <employee ID> <employee name> <Designation>
2. Search for the employee with their employee ID.
3. List the Id's of all the employees for a given input name.
4. For an inputted employee designation, list all the employee with the name and ID based on the joining date.

Requirement:

1. Implement the above problem statement using Python 3.7.
2. Create a BST data structure for storing employee details. The details should be read from a file **"PS11Q1.txt".** The program should create BST with a single node containing details such as <employee ID> <employee name> and < Designation>. Each employee details are in single line where the details are separated by ",". In the output file "**outputPS11Q1.txt"**, enter the total number of employee records that are created.
3. Search and list all the details of employees from the prompt file. The file **"promptsPS11Q1.txt"** contain the list of abc's employees IDs with a tag **"Search ID:"**. The program should be written to search the BST for each

employee ID from the same file and the corresponding details should be written into the file "**outputPS11Q1.txt"**. If an employee ID is not present in the BST, the program should write "not found" in the file.

4. Search for all employees for a given input name. The name to be searched will be provided with a tag **"Search Name:"** in the file **"promptsPS11.txt"** and Id no's of the employees matching with the given name should be output to the file "**outputPS11Q1.txt"**
5. List the employee name and ID as per the seniority for a prompted designation. Seniority is calculated based on the employee id. The employee with the lower employee id is more senior than the one with a higher employee id. The designation to be searched will be provided with a tag **"Search Designation:"** in the file **"promptsPS11Q1.txt"** The result including the employee name and employee id should be written to a file called **"outputPS11Q1.txt"**
6. Perform an analysis for the features above and give the running time in terms of input size: n.

Sample input files

Details of the employee should be provided using a comma separated list. One employee's details will be present per line.

Sample PS11Q1.txt

Rajesh Sharma, 1004, CEO
Jordan Leon, 1012, Project Manager
Mark Antony, 1006, CTO
Divya Rajesh, 1013, software Engg
Anvar Mohamed, 1017, Project Manager
Rajesh Nair, 1103, Software Engg
Najeem Ali, 1105, PRO
Sony Rajesh, 1015, Project Manager
Arun Kumar, 1009, Trainee

Sample promptsPS11Q1.txt

Search ID: 1004
Search ID: 1005
Search ID: 1006
Search ID: 1009
Search ID: 1105

Search Name: Rajesh

Search Designation: Project Manager

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

Sample output and output files

Below is the sample output after executing

Sample outputPS11Q1.txt

```
9 Binary Tree Created with the employee details (from file abc_emp_dataPS11.txt")
---------- Search by ID ----------
1004    Rajesh Sharma
1005  not found
1006    Mark Antony
1009    Arun Kumar
1105    Najeem Ali
-----------------------------------------------
---------- Search by Name: Rajesh -----------
        Rajesh Sharma
        Divya Rajesh
        Sony Rajesh
        Rajesh Nair
        -----------------------------------------------
        ----------List Employees by Designation: Project Manager ----------
        Jordan Leon, 1012
        Sony Rajesh, 1015
        Anvar Mohamed, 1017
        -----------------------------------------------
```

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

2. Deliverables

1. Word document **designPS11Q1_<student_id>.docx** detailing your design and time complexity of the algorithm.
2. **InputPS11Q1.txt and promptsPS11Q1.txt** is file used for testing
3. **OutputPS11Q1.txt** file generated while testing
4. **.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 file in a folder with the name:

[Student id]_PS11Q1.zip .

3. Problem Statement –Question 2

You are the leader of an elite task force that is commissioned to guard the borders of your country. You are allowed to build a team of your choice and each member can be equipped with one unique weapon. Each weapon fires a different type of ammunition and has a different capability of damage. The ammunition for these weapons come in packs of 100 units and have specific weights. Collectively, the team can carry a total weight of x kgs. As the leader of the task force it is your responsibility to decide the ratios of ammunition (full packs or partial packs) the team caries in order to maximize damage capability.

Requirements:

1. Formulate an efficient algorithm using Greedy Method to determine which ammunition to carry in what ratio to maximize damage capability.
2. Analyse the time complexity of your algorithm.
3. Implement the above problem statement using Python 3.7.

Sample Input:

For example, if there are 6 different team members carrying 6 weapons with a total ammunition carrying capacity of 118 kgs. Find the ratios in which each ammunition should be taken such as to maximize damage

Ammunition (i)	Weight per pack	Damage per pack
1	10	20
2	30	40
3	18	38
4	80	60
5	10	15
6	20	22

Input should be taken in through a file called "**inputPS11Q2.txt**" which has the fixed format mentioned below using the "/" as a field separator:

Weapons : <count>

MaxWeight : <weight>

<Ammunition i> / <Weight i> / <Damage i>

Ex:

Weapons : 6

MaxWeight : 118

A1 / 10 / 20

A2 / 30 / 40

A3 / 18 / 38

…

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:

Total Damage: 157.2

Ammunition Packs Selection Ratio:

A1 > 1

A2 > 1

A3 > 1

A4 > 0.37

A5 > 1

A6 > 1

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

Display the output in **outputPS11Q2.txt**.

5. Deliverables

1. Word document **designPS11Q2_<student id>.docx** detailing your design and time complexity of the algorithm.
2. **inputPS11Q2.txt** file used for testing
3. **outputPS11Q2.txt** file generated while testing
4. **.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 file in a folder with the name:

[Student id]_ PS11Q2.zip

6. 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.

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.

7. Deadline

1. The strict deadline for submission of the assignment is **<Refer E-learn Course page>.**

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.

8. How to submit

1. All the deliverables(zip files from section 2 and 5) must be combined in one zip file and named as **<Student ID>.zip**

2. Assignments should be submitted via E-Learn > Assignment section. Assignment submitted via other means like email etc. will not be graded.

9. Evaluation

1. The assignment carries **13 +12 =25 Marks**.
2. Grading will depend on
    i. Fully executable code with all functionality working as expected
    ii. Well-structured and commented code
    iii. 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. If two different groups submit the same code, both teams will get zero marks**.
7. Source code files which contain compilation errors will get at most 25% of the value of that question.

9. Readings

Text book: Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, WileyGoodrich, Roberto Tamassia, 2006, Wiley (Students Edition). **Chapters:** 2.3,5.1