Birla Institute of Technology & Science, Pilani

Work-Integrated Learning Programmes Division
MTech in Data Science & Engineering
S1_2022-2023, DSECLZG519- Data Structures & Algorithms Design

Assignment 2 – PS09 - BST Dictionary - [Weightage 13%]

Read through this entire document very carefully before you start!

1. Problem Statement

Python uses Hash-Tables for implementing Dictionaries.

You have to implement python-**dictionary** while using BST. But in this dictionary, the keys are numerical and values could be either string or numerical.

In this implementation **overload index-operator**([]). For example, Dictionary[2] will give you the value stored at key=2.

Also implement ".keys()" function and ".values()" like that of a dictionary.

BST-Dictionary keys are numerical but values could be a list, tuple, string, BST-Dictionary or any other primitive data types.

Make sure your searching is of order "log(n)".
Commands given in input file should be able to run exactly(see example for details)

Remember following points:

1. Keys are always greater than 0.

2. You can research __getitem__ for operator-overloading.

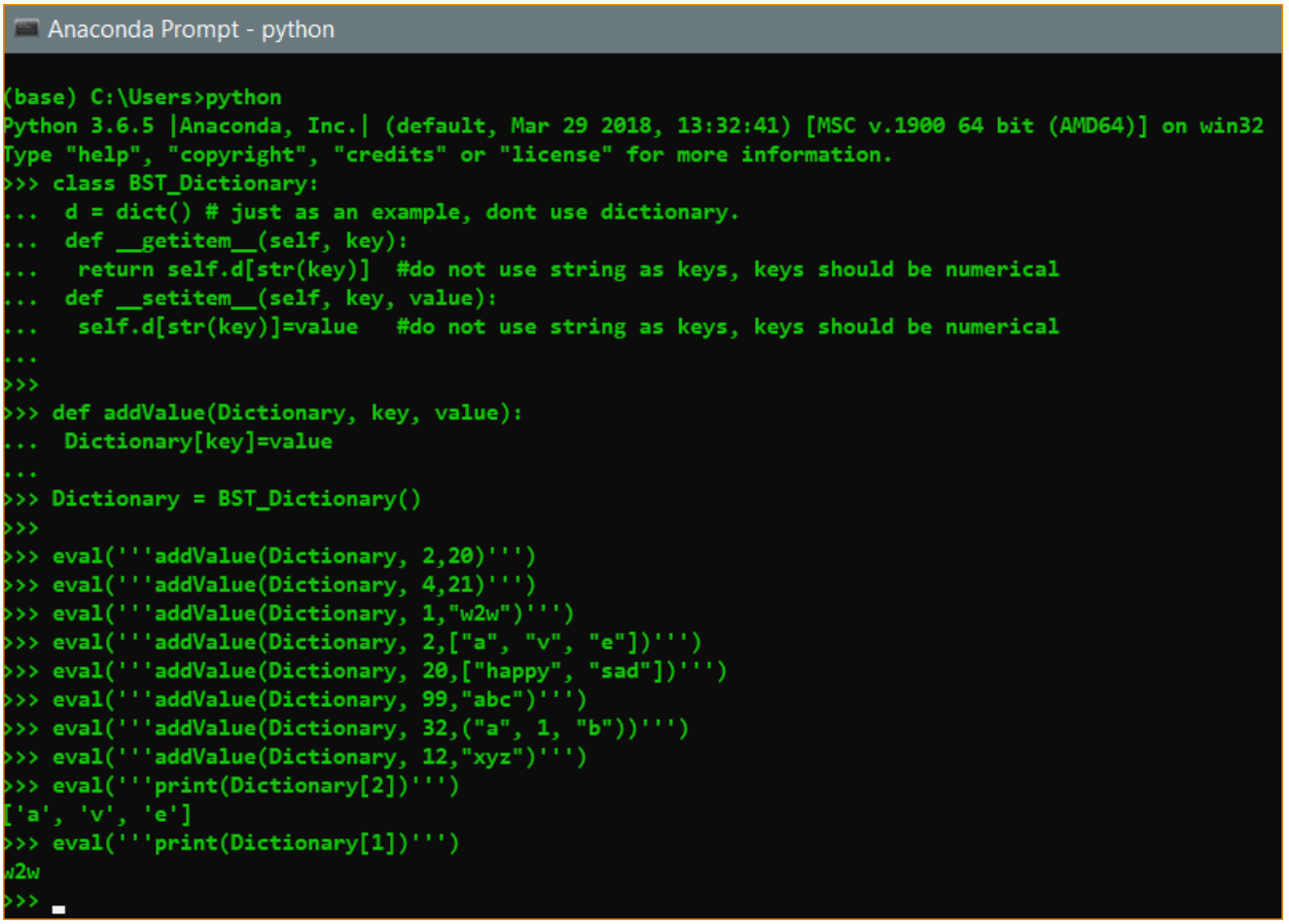
3. Do not use in-built "dictionary" in python anywhere for this assignment.

4. DO-NOT use dictionary anywhere in the code.

5. DO-NOT copy from any website, other groups might copy as well and then it would be a plagiarism case.

6. No more test-cases would be provided, you'd have to consider all possible cases/scenarios for fail-free execution.

Adding an Example to get you going..



.

Requirements

1. Implement the above problem statement using Python 3.7
2. Read the input from a file (inputPS09.txt) and Print the sequence of events and total time taken for the entire queue of processes to be executed.
3. You will output your answers to a file (outputPS09.txt) for each line.
4. Perform an analysis for the features above and give the running time in terms of input size: n along with justification.

Example Input:

Sample Input:

Input will be taken from the file**(inputPS09.txt**) .

Sample Input Example

```
addValue(Dictionary, 2,20)
addValue(Dictionary, 4,21)
addValue(Dictionary, 1,"w2w")
addValue(Dictionary, 2,["a", "v", "e"])
addValue(Dictionary, 20,["happy", "sad"])
addValue(Dictionary, 99,"abc")
addValue(Dictionary, 32,("a", 1, "b"))
addValue(Dictionary, 12,"xyz")
Dictionary.keys()
Dictionary.values()
print(Dictionary[2])
print(Dictionary[1])
```

Sample Output:

Display the output in **outputPS09.txt**.

```
2, 1, 4, 20, 12, 99, 32
[["a" ,"v", "e"], "w2w", 21, ["happy", "sad"], "xyz", "abc", ("a", 1, "b")]
["a", "v", "e"]
"w2w"
```

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

1. Deliverables

1. PDF document **designPS09_<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. **inputPS09.txt** file used for testing

4. **outputPS09.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_PS09.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. 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 modeling the problem with the cost implications.

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

10. You may ask queries in the dedicated discussion section. Beware that only hints will be provided and queries asked **in other channels will not be responded to**.

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 **Friday, 10th March 2023 11:55PM.**

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. Assignments

submitted via other means like email etc. will not be graded.

6. Evaluation

1. The assignment carries **13 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.
   d. Every bug in the functionality will have negative marking.
   e. Marks will be deducted if your program fails to read the input file used for evaluation due to change / deviation from the required syntax.
   f. Use of only native data types and avoiding libraries like numpy, graph and pandas will get additional marks.
3. 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 here.
4. 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 on <2,6>**

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