CMPE 224
Fall 2021
Programming Homework 5

This assignment is due by 23:59 on Wednesday, 5 January 2022.

You are welcome to ask your HW related questions. You should use only one of these options:

1. Moodle Homework **Question Forum**: HW Question-and-Answer (Q&A) Forum on Moodle is always available. Use the "Forum" link at the course Moodle page.
2. Homework **RECITATION HOURS**: There will be two Q&A RECITATION HOURs on the following days:
   - CMPE224-HW5-OfficeHour1: 24 Dec 2021, 06:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/96191547285
   - CMPE224-HW5-OfficeHour2: 29 Dec 2021, 06:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/92350752176
   - CMPE224-HW5-OfficeHour3: 3 Jan 2022, 06:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/98161641471

Note: Please make sure that you have read the HW document well before participating. However, no HW related questions will be accepted except from the above options.

PROGRAMMING TASK(s)

In this homework, you are expected to implement trie data structure and following functions.

First of all, you should create a trie data structure, then you should insert all words in the given txt files, read from user. Finally, you should complete following functions:

Boolean Search(String arg): This function should return true if given argument is in your trie, otherwise it should return false.

Void autoComplete(String prefix): This function should print all strings start with given prefix in your trie, lexicographically.

Void reverseAutoComplete(String suffix): This function should print all strings end with given suffix in your trie, lexicographically. For this function, you may consider using multi-trie solution, or research and use more complex data structures such as suffix arrays.

Void FullAutoComplete(String prefix, String suffix): This function should print all strings start with given prefix and end with given suffix in your trie, lexicographically. For this function, you may consider using multi-trie solution, or research and use more complex data structures such as suffix arrays.

Void findTopK(int k): This function should print top k words that have most occurrences, lexicographically. For this function, you may use a symbol table that keeps track of the number of appearances of each key.

Void SolvePuzzle(String filepath): This function should read input from the given filepath and print all possible words in your trie Example file,puzzle1.txt, is as follow:

```
w o r l d
a a d e f
l b t d e
l a d e r
a b d e r
```

(Each character in the given .txt file is separated with one space character.)

You can move horizontally, vertically, and diagonally in the given grid. You should find and print all valid words in your trie such as “world”, “water”, “wall” etc, lexicographically.

Example inputs and outputs:

Let input1.txt has following lines:

```
I ate dinner
We had a three course meal
Brad came to dinner with us
He loves fish tacos
In the end, we all felt like we ate too much
We all agreed; it was a magnificent evening
She is in a loveless relationship
We have wall
```

1-) Search

```
Input1.txt    (enter words file)
1             (represents search function)
Ate           (enter search word)
True
```

```
Input1.txt    (enter words file)
1             (represents search function)
had           (enter search word)
True
```

```
Input1.txt    (enter words file)
1             (represents search function)
world         (enter search word)
False
```

```
Input1.txt    (enter words file)
1             (represents search function)
money         (enter search word)
False
```

2-) AutoComplete

```
Input1.txt    (enter words file)
2             (represents autocomplete function)
a             (enter auto-completed word)
a, agreed, all, ate
```

```
Input1.txt    (enter words file)
2             (represents autocomplete function)
lov           (enter auto-completed word)
loveless, loves
```

```
Input1.txt    (enter words file)
2             (represents autocomplete function)
asds          (enter auto-completed word)
No words
```

3-) ReverseAutoComplete

```
Input1.txt    (enter words file)
3             (represents reverse autocomplete function)
s             (enter reverse auto-completed word)
is, loveless, loves, tacos, us, was
```

```
Input1.txt    (enter words file)
3             (represents reverse autocomplete function)
qx            (enter reverse auto-completed word)
No word
```

4-)FullComplete

```
Input1.txt    (enter words file)
4             (represents full autocomplete function)
th ee         (enter full auto-completed words, prefix and suffix respectively.)
three
```

```
Input1.txt    (enter words file)
4             (represents full autocomplete function)
thc ee        (enter full auto-completed words, prefix and suffix respectively.)
No word
```

5-) findTopK

```
Input1.txt    (enter words file)
5             (represents find top K function)
2             (enter number K)
a, we
```

6-) SolvePuzzle

```
Input1.txt    (enter words file)
6             (represents solve puzzle function)
 puzzle1.txt   (enter puzzle file)
a, all, ate, wall
```

WHAT TO HAND IN

A zip file for both parts containing:

- The Java sources for your program.
- The Java sources should be WELL DOCUMENTED as comments, as part of your grade will be based on the level of your comments.
- You should test your Java source files on (if) available Moodle VPL environment to ensure your code solution's correctness before submitting. VPL simply tests your program's output by checking against given sample input.
- A single **maximum-4 pages** PDF report document that explains your own answers for both tasks in a clearly readable PA report format (refer to **PA REPORT FORMAT** section).

PA REPORT FORMAT

A programming assignment report is a self-description of a programming assignment and your solution. The report must not be hand-written. You may use a word processor or the on-line editor of your choice and prepare as a PDF document. The report must be grammatically correct and use complete English sentences. Each report should include the following sections, in the order given:

Information (%5): This section includes your ID, name, section, assignment number information properly.

Problem Statement and Code Design (%30): Include a brief summary of the problem and/or your sub-tasks to be completed in this assignment. You should show your modular design rationale by creating a structure chart that indicates your top-down, stepwise refinement of the problem solution. You may create the structure chart using available graphical tools like MS PowerPoint, SmartDraw etc.

Implementation and Functionality (%40): Since you have modular source code, you should describe each sub-module (program) in this section. Each sub-module should include names and types of any input/output parameters as well as the pseudocode algorithm that used for completing its task. By this way, you give meaning to each chart boxes from the previous section.

Testing (%15): You should provide a tester class that is able to identify key test points of your program. This class should be able to generate additional (apart from the given sample input/output) test data for the purpose of being clear on what aspects of the solution are being tested with each set. This section should also include a description of any program *bugs* that is,

tests which has incorrect results. You should write these to describe your tests, summarize your results, and argue that they cover all types of program behavior.

Final Assessments (%10): In this final section, you should briefly answer the following questions:

- What were the trouble points in completing this assignment?
- Which parts were the most challenging for you?
- What did you like about the assignment? What did you learn from it?

IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:59 on Friday, January 5th.

2. You should upload your homework to Moodle before the deadline. No hardcopy submission is needed. You should upload files and any additional files if you wrote additional classes in your solution as a single archive file (e.g., zip, rar).

3. The standard rules about late homework submissions apply (20 points will be deducted for each late day). Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You ARE NOT ALLOWED to modify the given method names. However, if necessary, you may define additional data members and member functions.

5. Your classes' name MUST BE as shown in the homework description.

6. The submissions that do not obey these rules will not be graded.

7. To increase the efficiency of the grading process as well as the readability of your code, you have to follow the following instructions about the format and general layout of your program.

8. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//-------------------------------------------------
// Title: Scheduler tester class
// Author: Name/Surname
// ID: 2100000000
// Section: 1
// Assignment: 1
// Description: This class tests the …
//-------------------------------------------------
```

9. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to your classes, functions, declarations etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
void setVariable(char varName, int varValue)
//-------------------------------------------------------
// Summary: Assigns a value to the variable whose
// name is given.
// Precondition: varName is a char and varValue is an
// integer
// Postcondition: The value of the variable is set.
//-------------------------------------------------------
{
      // Body of the function
}
```

10. Indentation, indentation, indentation...

11. This homework will be graded by your TAs, Bedrettin Çetinkaya, Deniz Merve Gündüz and İbrahim İleri. Thus, you may ask them your homework related questions through HW forum on Moodle course page. You are also welcome to ask your course instructors Tolga Kurtuluş Çapın and Ulaş Güleç for help.