Computational Problem Solving AutoComplete

CSCI-603 Lab 5

1 Problem

Many of today's cell phones, text editors, and IDEs have an auto-complete feature. This feature attempts to complete the word for you using a list of known words to make suggestions.

Like with many auto-completion tools, you can cycle through a list of possible words with the given prefix. The system you will implement first sorts a list of words into lexicographic order. Given a prefix, it will make auto-complete suggestions starting with the first possible suggestion in the sorted list and - if asked - iterates through the list making further suggestions. When it reaches the last possible suggestions in the list it will restart back at the first suggestion.

Assume you have a file `test.dic` with the known words as follows:

```
baz
test
foo
two
zoo
three
bar
fool
dew
duel
guess
baby
boy
```

An example of the auto-complete system would be:

```
$ python3 auto_complete.py test.dic
The sorted list: ['baby', 'bar', 'baz', 'boy', 'duel', 'few',
'foo', 'fool', 'guess', 'test', 'three', 'two', 'zoo']
```

```
Welcome to Auto-complete!
Usage: Enter a prefix to auto-complete.
Entering nothing will print the first word in the sorted list.
Enter <QUIT> to exit.
Enter a prefix to search for:
baby
Enter a prefix to search for: app
No match
Enter a prefix to search for: ba
baby
Enter a prefix to search for:
bar
Enter a prefix to search for: f
few
Enter a prefix to search for: <QUIT>
Exiting Auto-complete! Good bye.
```

The sorted list of words was wrapped to fit on the page. It should be one line in your program.

2 Problem-solving Session

For the problem-solving session, you will work in groups of three to four students to answer the following questions. In order to receive full credit for the problem solving session you must be working for the full period or verify your work with either the instructor or SLI before leaving.

Assume you have a sorted list, `L`:

```
['aa', 'aaa', 'b', 'ba', 'bab', 'bba', 'bbb', 'bc', 'ca', 'cc']
```

1. Write a linear search algorithm, `findLastIndex`, that takes a sorted list of words, a prefix and a starting index (that matches the prefix), and finds the index of the *last* word that starts with the prefix.

2. If you were looking for words starting with `"bb"`, what starting index would you pass to `findLastIndex`? If `"bb"` was a word in the list, where would it be? Consider the same two questions for `"a"` and `"d"`.

3. Below is the code for binary search provided on this course.

```
1  def _binSearchRec(data, val, left, right):
2      """:return: The index of the first occurrence
3      of the value, if present, -1 otherwise. """
4      if left > right:
5          return -1
6      midindex = (left+right)//2
7      if data[midindex] == val:
8          return midindex
9      if data[midindex] > val:
10         return _binSearchRec(data, val, left, midindex-1)
11     else:
12         return _binSearchRec(data, val, midindex+1, right)
13
14 def binSearch(data, val):
15     return _binSearchRec(data, val, 0, len(data)-1)
```

   Modify this code to return the index you should pass to `findLastIndex` function to start looking for word matches. Work through an example using the above list with the prefix `"bb"`. You will be implementing this algorithm in your lab, so you should also consider edge cases. E.g. a prefix ordered before every element in the list (`"a"`), after every element (`"d"`), etc.

   Note: You are not required to write the entire function from scratch, you just need to indicate what are the changes.

4. Write a function that repeatedly asks the user for input until the user types and enters the string `"<QUIT>"`. It should print the word the user enters.