Data Structures

Trie Homework 3

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Problem #1: Find all substrings

- **def** list_substrs(long_str, queries_lst)
- An external function that takes a string **long_str** of length S and several queries (each of maximum length L).
- Return all query words that are substrings in **long_str**
  - S is very large and L is moderate length
  - Output order doesn't matter
- Find 2 solutions based on trie
  - Normal thinking: A trie based on str
  - Reverse thinking: A trie based on queries
  - Compare the time complexity
- Input: "hey**ab**cd**t**w**xy**w" and queries: "xy", "ab", "t", "yz"
  - Only print: "xy", "ab", "t"

```python
if __name__ == '__main__':
    long_str = 'heyabcdtwxyw'
    queries_lst = ["xy", "ab", "t", "yz"]

    ans = list_substrs(long_str, queries_lst)

    print(ans)
    # ['xy', 'ab', 't']
```

Problem #2: LeetCode 745 - Prefix and Suffix Search

- **def** f(self, prefix, suffix)
- Given a dictionary, and the queries above (prefix and suffix), find the word index that **starts with this prefix**, and **ends with this suffix**.
- If there are several matches, return the **last** index. -1 for no match.
- Assume dictionary: [“aae”, “apple”, “bannana”]
- Query [“a”, “e”] : it matches both “aae”, “apple” ⇒ return 1 [largest idx]
- Query [“a”, “x”] : ⇒ -1
- Query [“a”, “ae”] ⇒ 0
- Tip: there are several trie-based solutions

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."