Oursky Developer Pre-Test 2021

Thank you for applying to Oursky. When considering applicants for developer positions, we emphasize on their true technical skills, which can't be measured by their CV alone. That's why we hope you would take time to complete this pre-test.

Please do not disclose the content of this test to others.

For the answers, please pick a programming language you're familiar with.

To answer the questions, please create a gist link at gist.github.com, with 3 files named answer[1-3].md (or the correct extension for the programming language, e.g. answer1.c / answer2.md / answer3.py), and reply our pre-test interview email with the gist link. The gist link is expect to be configured with **non-searchable**. Also, please don't send Github repo or zip file to us.

All 3 questions are mandatory.

Your answer will be evaluated based on:

- Correctness of your solutions
- Cleanness of your code
- Expressiveness and Clarity of your logic.

Question 1

Write a function that takes two arrays as input, each array contains a list of A-Z; Your program should return True if the 2nd array is a subset of 1st array, or False if not.

For example:
isSubset([A,B,C,D,E], [A,E,D]) = true
isSubset([A,B,C,D,E], [A,D,Z]) = false
isSubset([A,D,E], [A,A,D,E]) = true

Please explain the computational complexity of your answer in Big-O notation, i.e. O(log n) or O(n ^2)?

Question 2

Design and implement a data structure for cache.

- get(key) - Get the value of the key if the key exists in the cache, otherwise return -1
- put(key, value, weight) - Set or insert the value, when the cache reaches its capacity, it should invalidate the least scored key. The score is calculated as:
  - when current_time != last_access_time: `weight / ln(current_time - last_access_time + 1)`
  - else: `weight / -100`

Your data structure should be optimized for the computational complexity of get(key) i.e. Average case for computational complexity of get(key) could be O(1).

In your code, you can assume common data structure such as array, different type of list, hash table are available.

Please explain the computational complexity of get(key) and put(...) in Big-O notation.

Question 3

Please understand the following program

```
function recur(n, cur) {
  if (!cur) {
      cur = 0;
  }
  if (n < 2) {
    throw new Error('Invalid input');
  }
  if (n === 2) {
    return 1 / n + cur;
  }
  return recur(n - 1, cur + 1 / (n * (n -1)));
}
```

- To prevent an infinite loop in a production system. Write a program doing the same calculation without recursion. Please note that a while loop is also not good in a production system.