### 题目描述

给定一个字符串，把字符串按照大写在前小写在后排序，输出排好后的第 K 个字母在原来字符串的索引。

相同字母输出第一个出现的位置。

### 输入描述

无

### 输出描述

无

### 用例

| | |
|---|---|
| 输入 | hAkDAjByBq 4 |
| 输出 | 6 |
| 说明 | 排好序后 AABBDhjkqy，第 4 个是 B，第一个出现的在原字符串 6 这个位置。 (注：索引是从 0 开始) |

### 题目解析

简单的排序，以及 字符串操作 考察。

用例中的排序规则其实就是 字典序排序 。

## JavaScript算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

rl.on("line", (line) => {
  const [str, idx] = line.split(" ");

  const ele = [...str].sort()[idx - 1];

  console.log(str.indexOf(ele));
});
```

## Java算法源码

```java
import java.util.Arrays;
import java.util.Scanner;

public class Main {
  // 输入获取
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    String str = sc.next();
    int idx = sc.nextInt();

    System.out.println(getResult(str, idx));
  }

  // 算法入口
  public static int getResult(String str, int idx) {
    char[] sArr = str.toCharArray();
    Arrays.sort(sArr);
    char target = sArr[idx - 1];
    return str.indexOf(target);
  }
}
```

## Python算法源码

```python
# 输入获取
s, i = input().split()


# 算法入口
def getResult():
    sArr = list(s)
    sArr.sort()

    tar = sArr[int(i) - 1]
    return s.find(tar)


# 算法调用
print(getResult())
```