11월 24일 (수)

1. 엘리스 알고리즘 문제 풀이 - 재귀호출

- k번째 수 찾기

k번째 수 찾기

n개의 숫자가 차례대로 주어질 때, 매 순간마다 "지금까지 입력된 숫자들 중에서 k번째로 작은 수"를 반환하는 프로그램을 작성하세요.

프로그램의 입력으로는 첫째줄에 n과 k가 입력되고, 둘째줄에 n개의 숫자가 차례대로 주어집니다.

입력 예시

```
10 3
1 9 8 5 2 3 5 6 2 10
```

Copy

출력 예시

```
-1 -1 9 8 5 3 3 3 2 2
```

Copy

문제 조건

- n은 100보다 작은 숫자입니다.
- 매 순간마다 지금까지의 입력중 k번째로 작은 수를 출력하되, 없다면 -1을 출력합니다.

입출력 예시 설명

10개의 숫자가 차례대로 주어집니다. 맨 처음 1만 입력을 받았을 경우, 3번째로 작은 숫자가 없으므로 -1을 출력합니다. 그 다음 9도 마찬가지입니다. 세 번째로 숫자 8을 입력받는 순간, 지금까지 입력받은 숫자는 1, 9, 8 세 개이고, 이 중 3 번째로 작은 숫자인 9를 출력합니다. 마찬가지로 숫자 하나를 입력받을 때 마다 3번째로 작은 숫자를 출력합니다.

```
def findKth(myInput, k) :
    result = []
    data = []

    for element in myInput:
        data.append(element)
        data.sort()

        if len(data) < k:
            result.append(-1)
        else:
            result.append(data[k-1])

    return result


def main():
```

```
    firstLine = [int(x) for x in input("n과 k를 입력하세요 (예시:10 3): ").split()]
    myInput = [int(x) for x in input("n개의 숫자를 차례대로 입력하세요 (예시:1 9 8 5 2 3 5 6 2 10): ").split()]

    print('정렬 결과: ', *findKth(myInput, firstLine[1]))
if __name__ == "__main__":
    main()
```

- **퀵 정렬 구현하기**

quick sort

입력으로 n개의 수가 주어지면, quick sort를 구현하는 프로그램을 작성하세요.

입력 예시

```
10 2 3 4 5 6 9 7 8 1
```

Copy

출력 예시

```
1 2 3 4 5 6 7 8 9 10
```

Copy

```
def quickSort(data):
    if len(data) <= 1:
        return data

    pivot = data[0]

    left = getSmallNumbers(data[1:], pivot)
    right = getLargeNumbers(data[1:], pivot)

    return quickSort(left) + [pivot] + quickSort(right)

def getSmallNumbers(data, pivot):
    arr = []
    for i in data:
        if i <= pivot:
            arr.append(i)
    return arr

def getLargeNumbers(data, pivot):
    arr = []
    for i in data:
        if i > pivot:
            arr.append(i)
    return arr


def main():
    line = [int(x) for x in input("정렬할 수를 입력하세요 (예시:10 2 3 4 5 6 9 7 8 1): ").split()]

    print('정렬 결과:', *quickSort(line))

if __name__ == "__main__":
    main()
```

- **올바른 괄호인지 판단하기**

올바른 괄호인지 판단하기

본 문제에서는 입력으로 주어지는 괄호가 올바른 괄호인지를 판단하는 프로그램을 작성합니다.

예를 들어, '(())' 은 올바른 괄호이지만, '(()))', 혹은 '(()()(' 는 올바른 괄호가 아닙니다.

올바른 괄호일때 'YES'를, 올바르지 않은 괄호일때 'NO'를 출력해 봅시다.

입력 예시 1

```
(())()
```

출력 예시 1

```
YES
```

입력 예시 2

```
((((())()))(()())(((())()))
```

출력 예시 2

```
YES
```

```
(())()()
```

Copy

출력 예시 3

```
NO
```

Copy

입력 예시 4

```
(((()))(()()))(())
```

Copy

출력 예시 4

```
NO
```

Copy

입력

괄호 p가 주어집니다.

출력

p가 올바른 괄호이면 YES, 그렇지 않으면 NO를 출력합니다.

```
def checkParen(p):
    '''
```

```
    괄호 문자열 p의 쌍이 맞으면 "YES", 아니면  "NO"를 반환
    1. 기저조건 처리
    2. p에서 인접한 괄호쌍을 찾아 제거
    3. checkParen()
    '''
    if len(p) == 0:
        return "YES"
    elif len(p) == 1:
        return "NO"

    for i in range(len(p)-1):
        if p[i] == '(' and p[i+1] == ')':
            q = p[:i] + p[i+2:]
            return checkParen(q)

    return "NO"

def main():
    '''
    이 부분은 수정하지 마세요.
    '''

    x = input()
    print(checkParen(x))

if __name__ == "__main__":
    main()
```