공간 복잡도란?

- 입력값과 문제를 해결하는 데 걸리는 공간과의 상관관계를 말합니다.

입력값이 2배로 늘어났을 때 문제를 해결하는 데 걸리는 공간은 몇배로 늘어나는지를 보는 것입니다.

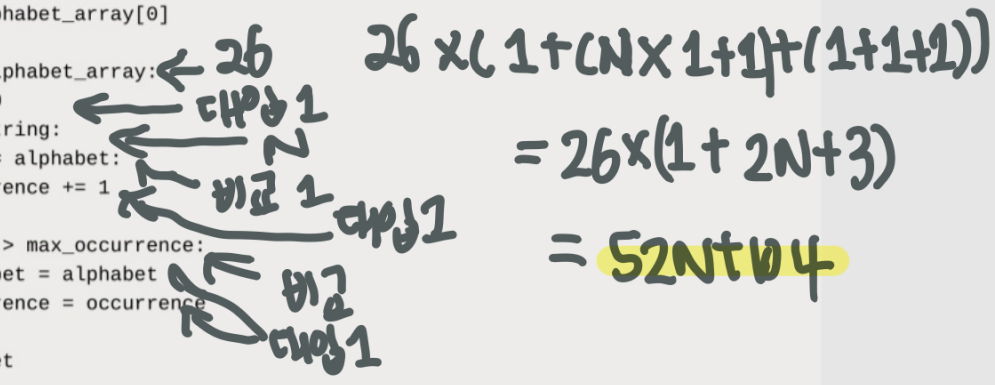
- 입력값이 늘어나도 걸리는 공간이 덜 늘어나는 알고리즘이 좋은 알고리즘 입니다.

```
input = "hello my name is sparta"


def find_max_occurred_alphabet(string):
    alphabet_array = ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n",
    max_occurrence = 0
    max_alphabet = alphabet_array[0]

    for alphabet in alphabet_array:
        occurrence = 0
        for char in string:
            if char == alphabet:
                occurrence += 1

        if occurrence > max_occurrence:
            max_alphabet = alphabet
            max_occurrence = occurrence

    return max_alphabet

result = find_max_occurred_alphabet(input)
print(result)
```

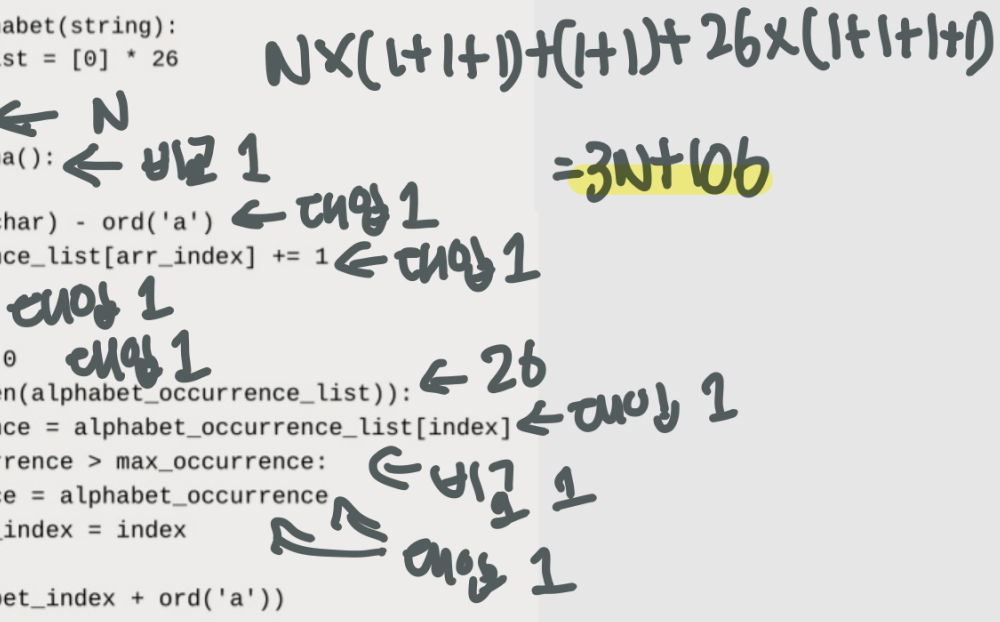
```
input = "hello my name is sparta"


def find_max_occurred_alphabet(string):
    alphabet_occurrence_list = [0] * 26

    for char in string:
        if not char.isalpha():
            continue
        arr_index = ord(char) - ord('a')
        alphabet_occurrence_list[arr_index] += 1

    max_occurrence = 0
    max_alphabet_index = 0
    for index in range(len(alphabet_occurrence_list)):
        alphabet_occurrence = alphabet_occurrence_list[index]
        if alphabet_occurrence > max_occurrence:
            max_occurrence = alphabet_occurrence
            max_alphabet_index = index

    return chr(max_alphabet_index + ord('a'))


result = find_max_occurred_alphabet(input)
print(result)
```

# 시간복잡도

```python
input = "hello my name is sparta"


def find_max_occurred_alphabet(string):
    alphabet_array = ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n",
    max_occurrence = 0
    max_alphabet = alphabet_array[0]

    for alphabet in alphabet_array:
        occurrence = 0
        for char in string:
            if char == alphabet:
                occurrence += 1

        if occurrence > max_occurrence:
            max_alphabet = alphabet
            max_occurrence = occurrence

    return max_alphabet

result = find_max_occurred_alphabet(input)
print(result)
```




```python
input = "hello my name is sparta"


def find_max_occurred_alphabet(string):
    alphabet_occurrence_list = [0] * 26

    for char in string:
        if not char.isalpha():
            continue
        arr_index = ord(char) - ord('a')
        alphabet_occurrence_list[arr_index] += 1

    max_occurrence = 0
    max_alphabet_index = 0
    for index in range(len(alphabet_occurrence_list)):
        alphabet_occurrence = alphabet_occurrence_list[index]
        if alphabet_occurrence > max_occurrence:
            max_occurrence = alphabet_occurrence
            max_alphabet_index = index

    return chr(max_alphabet_index + ord('a'))


result = find_max_occurred_alphabet(input)
print(result)
```




N^2 > 52N + 104 > 3N + 106

공간복잡도 보다는 시간 복잡도에 더 신경을 써야 한다