Chapter. 06

기본 탐색 알고리즘

기초 문제풀이

FAST CAMPUS
ONLINE
유형별 문제풀이

강사. 나동빈

Chapter. 06

기본 탐색 알고리즘(기초 문제풀이)

FAST CAMPUS
ONLINE
나동빈 강사.

I 혼자 힘으로 풀어 보기

문서 검색: https://www.acmicpc.net/problem/1543

문제 제목: 문서 검색

문제 난이도: 하(Easy)

문제 유형: 탐색

추천 풀이 시간: 20분

문서 검색: https://www.acmicpc.net/problem/1543

I 문제 풀이 핵심 아이디어

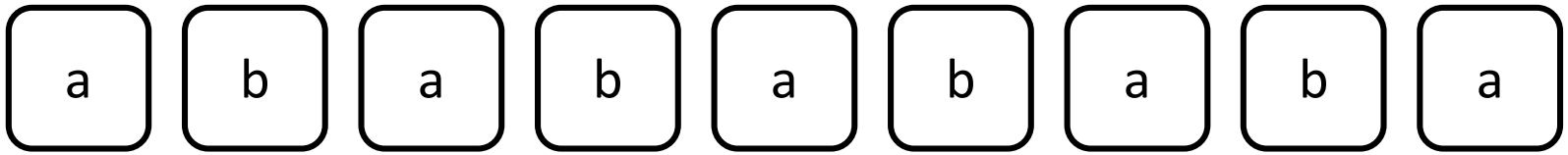
- 문서의 길이는 최대 2,500이고 단어의 길이는 최대 50입니다.
- 단순히 모든 경우의 수를 계산하여 문제를 해결할 수 있습니다.
- 시간 복잡도 $O(NM)$의 알고리즘으로 해결할 수 있습니다.

문서 검색: https://www.acmicpc.net/problem/1543

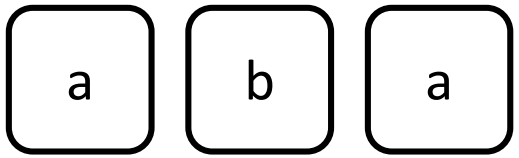
I 문제 풀이 핵심 아이디어

- 문서와 단어의 위치를 맞추어서 반복적으로 비교합니다.

[문서]



[단어]

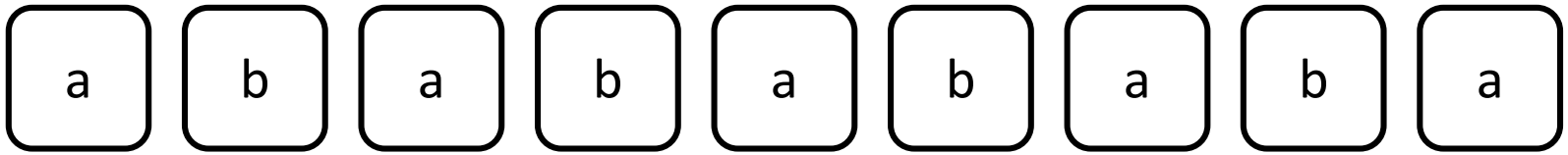


문서 검색: https://www.acmicpc.net/problem/1543

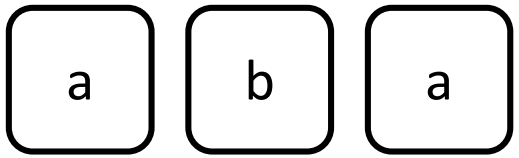
I 문제 풀이 핵심 아이디어

- 문서와 단어의 위치를 맞추어서 반복적으로 비교합니다.

[문서]



[단어]



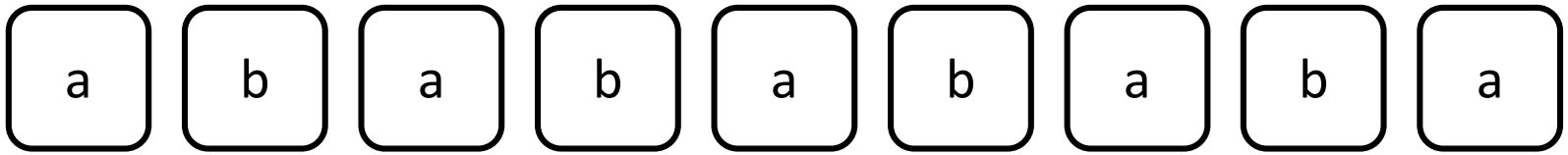
결과: 0

문서 검색: https://www.acmicpc.net/problem/1543

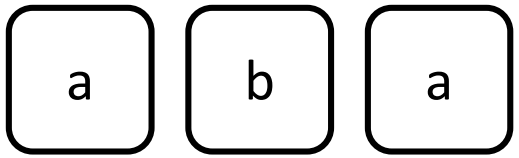
문제 풀이 핵심 아이디어

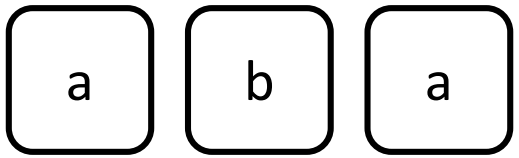
- 문서와 단어의 위치를 맞추어서 반복적으로 비교합니다.

[문서]



[단어]





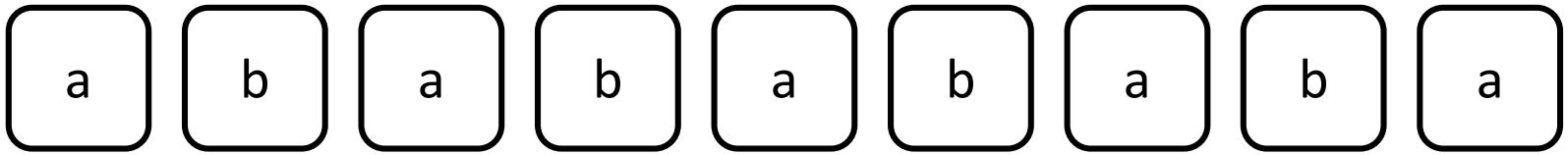
결과: 1

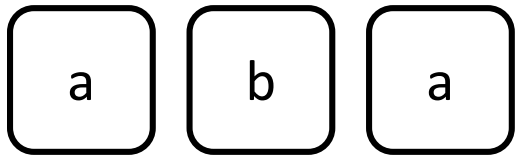
문서 검색: https://www.acmicpc.net/problem/1543

I 문제 풀이 핵심 아이디어

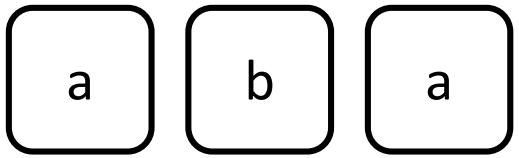
- 문서와 단어의 위치를 맞추어서 반복적으로 비교합니다.

[문서]





[단어]



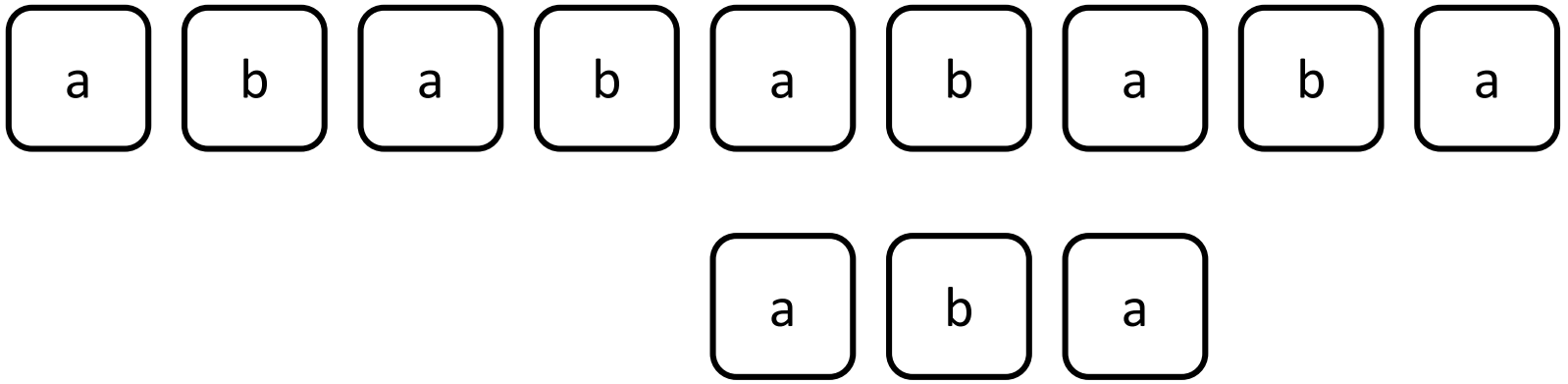
결과: 1

문서 검색: https://www.acmicpc.net/problem/1543

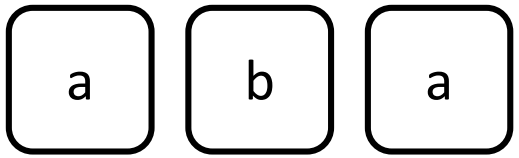
I 문제 풀이 핵심 아이디어

- 문서와 단어의 위치를 맞추어서 반복적으로 비교합니다.

[문서]



[단어]



결과: 2

문서 검색: https://www.acmicpc.net/problem/1543

소스코드

```
document = input()
word = input()

index = 0
result = 0

while len(document) - index >= len(word):
    if document[index:index + len(word)] == word: # 문서에서 보고 있는 단어가 찾고자 하는 단어인 경우
        result += 1
        index += len(word)
    else:
        index += 1

print(result)
```

I 혼자 힘으로 풀어 보기

새: https://www.acmicpc.net/problem/1568

문제 제목: 새

문제 난이도: 하(Easy)

문제 유형: 탐색

추천 풀이 시간: 20분

새: https://www.acmicpc.net/problem/1568

I 문제 풀이 핵심 아이디어

- N이 최대 1,000,000,000입니다.
- K가 반복적으로 증가하므로, 날아가는 새의 마리 수는 빠르게 증가합니다.
- 따라서 문제에서 요구하는 대로 단순히 구현하여 정답 처리를 받을 수 있습니다.

새: https://www.acmicpc.net/problem/1568

| 소스코드

```python
n = int(input())
result = 0
k = 1

while n != 0: # 모든 새가 날아갈 때까지
    if k > n:
        k = 1
    n -= k
    k += 1
    result += 1

print(result)
```

베스트셀러: https://www.acmicpc.net/problem/1302

혼자 힘으로 풀어 보기

문제 제목: 베스트셀러

문제 난이도: 하(Easy)

문제 유형: 탐색

추천 풀이 시간: 20분

베스트셀러: https://www.acmicpc.net/problem/1302

I 문제 풀이 핵심 아이디어

- 본 문제는 가장 많이 등장한 문자열을 출력하는 문제와 동일합니다.
- 등장 횟수를 계산할 때는 파이썬의 Dictionary 자료형을 이용하면 효과적입니다.

베스트셀러: https://www.acmicpc.net/problem/1302

소스코드

```
n = int(input())

books = {}

for _ in range(n):
    book = input()
    if book not in books:
        books[book] = 1
    else:
        books[book] += 1

target = max(books.values())
array = []

for book, number in books.items():
    if number == target:
        array.append(book)

print(sorted(array)[0])
```

I 혼자 힘으로 풀어 보기

트로피 진열: https://www.acmicpc.net/problem/1668

문제 제목: 트로피 진열

문제 난이도: 하(Easy)

문제 유형: 탐색

추천 풀이 시간: 20분

트로피 진열: https://www.acmicpc.net/problem/1668

I 문제 풀이 핵심 아이디어

- 선반 위에 올려져 있는 트로피들에 대하여 왼쪽에서, 오른쪽에서 봤을 때 보이는 트로피의 수를 각각 구합니다.
- 트로피의 개수 N이 최대 50이므로 단순히 구현하면 됩니다.

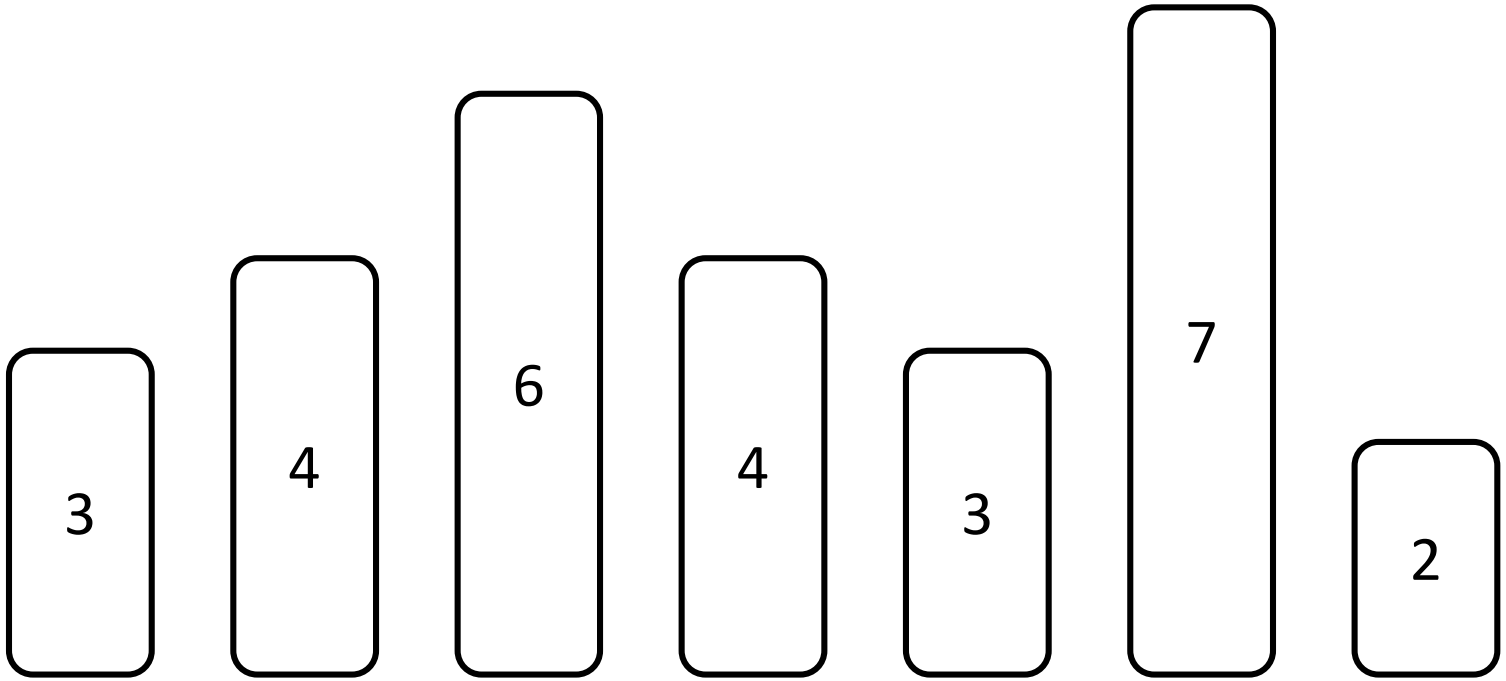
트로피 진열: https://www.acmicpc.net/problem/1668

I 문제 풀이 핵심 아이디어

- 선반 위에 올려져 있는 트로피들에 대하여 왼쪽에서, 오른쪽에서 봤을 때 보이는 트로피의 수를 각각 구합니다.
- 트로피의 개수 N이 최대 50이므로 단순히 구현하면 됩니다.

왼쪽: 4

오른쪽: 2



트로피 진열: https://www.acmicpc.net/problem/1668

| 소스코드

```python
def ascending(array):
    now = array[0]
    result = 1
    for i in range(1, len(array)):
        if now < array[i]:
            result += 1
            now = array[i]
    return result

n = int(input())
array = []

for _ in range(n):
    array.append(int(input()))

print(ascending(array))
array.reverse()
print(ascending(array))
```

I 혼자 힘으로 풀어 보기

성 지키기: https://www.acmicpc.net/problem/1236

문제 제목: 성 지키기

문제 난이도: 하(Easy)

문제 유형: 탐색

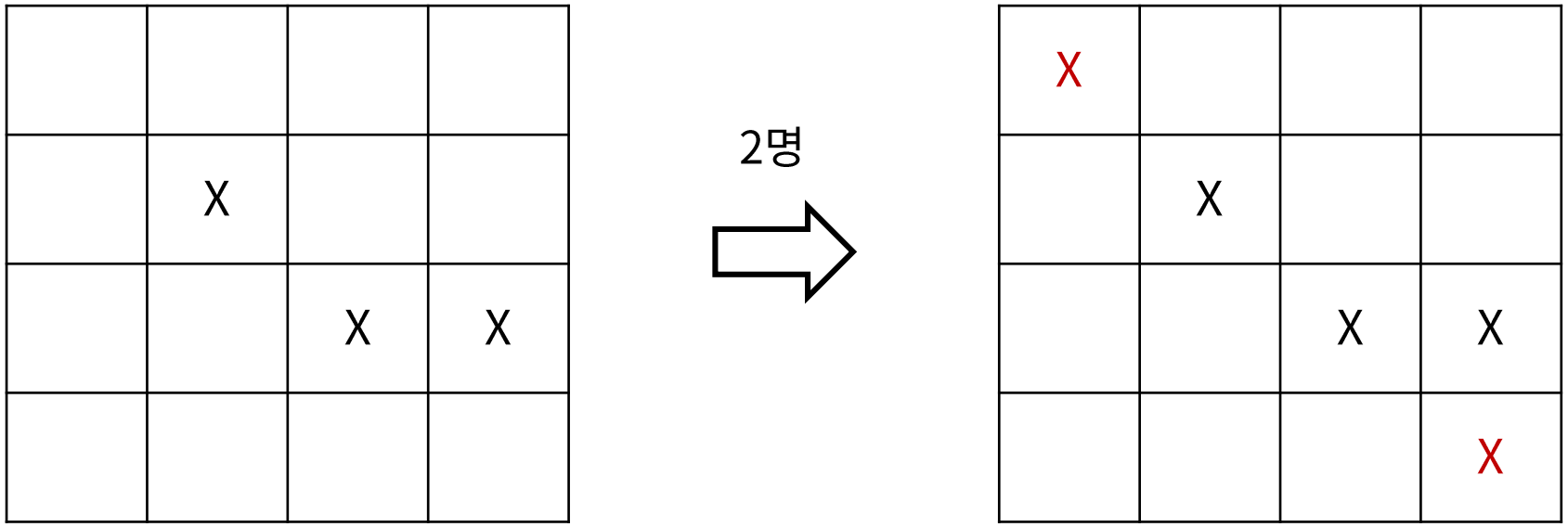
추천 풀이 시간: 20분

성 지키기: https://www.acmicpc.net/problem/1236

I 문제 풀이 핵심 아이디어

- 모든 행과 모든 열에 한 명 이상의 경비원이 있어야 합니다.
- 행 기준·열 기준으로 필요한 경비원의 수를 각각 계산하여 더 큰 수를 출력합니다.

[예시]

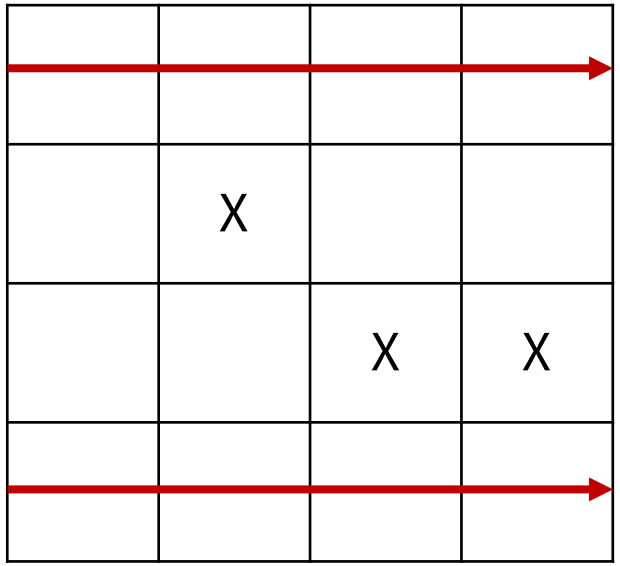


성 지키기: https://www.acmicpc.net/problem/1236

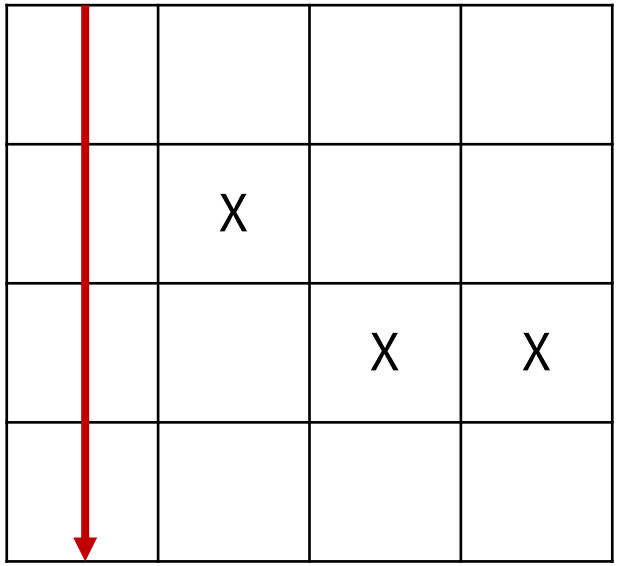
문제 풀이 핵심 아이디어

- 모든 행과 모든 열에 한 명 이상의 경비원이 있어야 합니다.
- 행 기준·열 기준으로 필요한 경비원의 수를 각각 계산하여 더 큰 수를 출력합니다.

행: 2



열: 1



성 지키기: https://www.acmicpc.net/problem/1236

| 소스코드

```python
n, m = map(int, input().split())
array = []

for _ in range(n):
    array.append(input())

row = [0] * n
column = [0] * m

for i in range(n):
    for j in range(m):
        if array[i][j] == 'X':
            row[i] = 1
            column[j] = 1

row_count = 0
for i in range(n):
    if row[i] == 0:
        row_count += 1

column_count = 0
for j in range(m):
    if column[j] == 0:
        column_count += 1

print(max(row_count, column_count))
```