알고리즘 Week 7

19기 정규세션

TOBIG’S 18기 지윤혁

Contents

Unit 01 | 5주차 과제 리뷰

Unit 02 | 정렬 및 분할정복

Unit 03 | 정렬 알고리즘 소개

Unit 03 | 7주차 과제 소개

Unit 01 | 5주차 과제 리뷰

19기 정규세션
TOBIG'S 18기 지윤혁

문제1. 1, 2, 3 더하기 (DP)

```python
# 백준 9095 1, 2, 3 더하기

import sys
input = sys.stdin.readline
# sys.stdin = open("input.txt", "r")

dp = [0]*11
dp[1], dp[2], dp[3] = 1, 2, 4

for i in range(4, 11):
    # dp[n] = dp[n-3] + dp[n-2] + dp[n-1]
    dp[i] = sum(dp[i-3:i])

t = int(input().rstrip())
for _ in range(t):
    n = int(input().rstrip())
    print(dp[n])
```

점화식 dp[n] = dp[n-3] + dp[n-2] + dp[n-1]

19기 정규세션
TOBIG'S 18기 지윤혁

문제1. 1, 2, 3 더하기 (재귀)

```python
# B9095 1,2,3 더하기

import sys
input = sys.stdin.readline

t = int(input().rstrip())
n_list = list(range(1, 4))
temp = []
temp_list = []

def solution(sum_tot, sum_val):
    if sum_val == sum_tot:
        temp_list.append(temp)
        return
    elif sum_val > sum_tot:
        return
    for i in range(len(n_list)):
        temp.append(n_list[i])
        solution(sum_tot, sum_val+n_list[i])
        temp.pop()

for _ in range(t):
    n = int(input().rstrip())
    solution(n, 0)
    print(len(temp_list))
    temp_list.clear()
```

② ③ ①

n_list: 1, 2, 3이 포함된 리스트

temp: n을 만들 수 있는 경우를 담은 각각의 순열

temp_list: n을 만들 수 있는 모든 경우를 담는 리스트

① n_list 내에 있는 1, 2, 3을 중복을 포함하여 선택

② temp에 포함된 값들의 합이 n과 일치하면 stop

③ 가지치기: temp에 포함된 값들이 n보다 클 경우도 stop

19기 정규세션
TOBIG'S 18기 지윤혁

문제2. 1로 만들기 (Bottom-Up 방식)

```python
# 백준 1463 1로 만들기

import sys
input = sys.stdin.readline
# sys.stdin = open("input.txt", "r")

n = int(input().rstrip())
dp = [0]*(n+1)

① for i in range(2, n+1):
②     dp[i] = dp[i-1] + 1
      if i%3 == 0:
③         dp[i] = min(dp[i], dp[i//3]+1)
      if i%2 == 0:
          dp[i] = min(dp[i], dp[i//2]+1)

print(dp[n])
```

n을 1로 만들기= 1을 n으로 만들기

① 2부터 n 까지 만드는 연산을 반복

② 1을 더하는 연산을 우선 시행

③ 2 또는 3으로 나누어 떨어진다면,
1을 빼서 만드는 것보다 작은지 비교

④ 최종적으로 n을 만드는데 필요한 최소 연산 출력

19기 정규세션
TOBIG'S 18기 지윤혁

문제3. 계단 오르기

```python
# 백준 2579 계단 오르기

import sys
input = sys.stdin.readline
# sys.stdin = open("input.txt", "r")

n = int(input().rstrip())
n_list = [int(input().rstrip()) for _ in range(n)]

dp = [0]*n

if n <= 2:
    print(sum(n_list))
else:
    dp[0], dp[1] = n_list[0], n_list[0] + n_list[1]

    for i in range(2, n):
        dp[i] = max(dp[i-3]+n_list[i-1]+n_list[i], dp[i-2]+n_list[i])

    print(dp[n-1])
```

① n번째 계단까지 올라가는 최댓값 담는 리스트

② 계단이 2개보다 작을 경우에는 모든 경우 더하여 출력

③ 1층과 2층의 경우는 최댓값을 직접 계산하여 입력

④ 1칸 올라가는 경우와 2칸 올라가는 경우의 최댓값 비교

⑤ n번째 계단까지 올라가는 최댓값 출력

19기 정규세션
TOBIG'S 18기 지윤혁

문제4. 포도주 시식

```python
# 백준 2156 포도주 시식

import sys
input = sys.stdin.readline
# sys.stdin = open("input.txt", "r")

n = int(input().rstrip())
n_list = [int(input().rstrip()) for _ in range(n)]

dp = [0]*n

if n < 3:
    print(sum(n_list))
else:
    dp[0], dp[1] = n_list[0], (n_list[0] + n_list[1])
    for i in range(2, n):
        dp[i] = max(dp[i-3]+n_list[i-1]+n_list[i], dp[i-2]+n_list[i])
        dp[i] = max(dp[i-1], dp[i])
    print(dp[n-1])
```

① 계단 오르기와 유사한 방식
이틀 연속으로 포도주를 마시지 않는 방법도 존재
이전값과 비교해서 최대값을 추출하는 부분을 추가

19기 정규세션
TOBIG'S 18기 지윤혁

문제5. 동전1

```
# 백준 2293 동전 1

import sys
input = sys.stdin.readline
# sys.stdin = open("input.txt", "r")

n, k = map(int, input().rstrip().split())
① n_list = [int(input().rstrip()) for _ in range(n)]
② dp = [0]*(k+1)
③ dp[0] = 1

④ for coin in n_list:
    for i in range(coin, k+1):
⑤       value = dp[i-coin]
        dp[i] += value
print(dp[k])
```

① n_list: 동전의 종류

② dp: 각 인덱스에 해당하는 금액을 만들기 위해 필요한 동전의 수

③ dp[0] = dp[k-k]: k가 되게 만드는 k원 동전의 갯수

④ 각각의 동전으로 만들 수 있는 금액을 계산

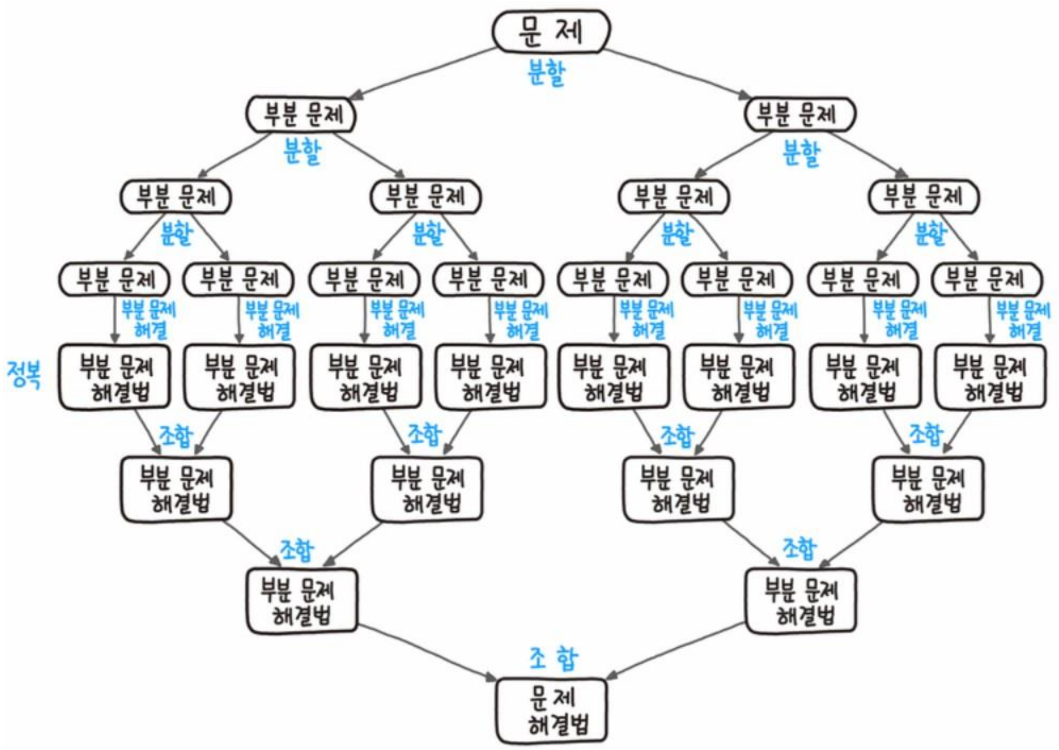
⑤ 이전의 동전으로 진행했던 부분 + 새로운 동전으로 만들 수 있는 방법

이전에 동전으로 만들었던 부분을 새로운 코인으로 만들 수 있는 만큼 추가

Unit 02 | 정렬 및 분할정복

분할정복 (Divide and Conquer)

주어진 문제가 간단한 문제가 될 때까지 문제를 재귀적으로 나눈 다음에
각 문제의 결과를 조합하여 전체 문제의 답을 계산하는 방식



분할 정복의 과정

1. 분할: 문제를 동일한 유형의 여러 하위 문제로 나눈다.
2. 정복: 가장 작은 단위의 하위 문제를 해결하여 정복한다.
3. 조합: 하위 문제에 대한 결과를 원래 문제에 대한 결과로 조합한다.

19기 정규세션
TOBIG'S 18기 지윤혁

장점

문제를 나눔으로써 어려운 문제를 해결할 수 있게 된다!

단점

함수를 재귀적으로 호출한다는 점에서 함수 호출로 인한 오버헤드가 발생하며,

스택에 다양한 데이터를 보관하고 있어야 하므로 스택 오버플로우가 발생하거나

과도한 메모리 사용을 하게 된다.

※오버헤드: 어떤 처리를 하기 위해 들어가는 간접적인 처리 시간 · 메모리

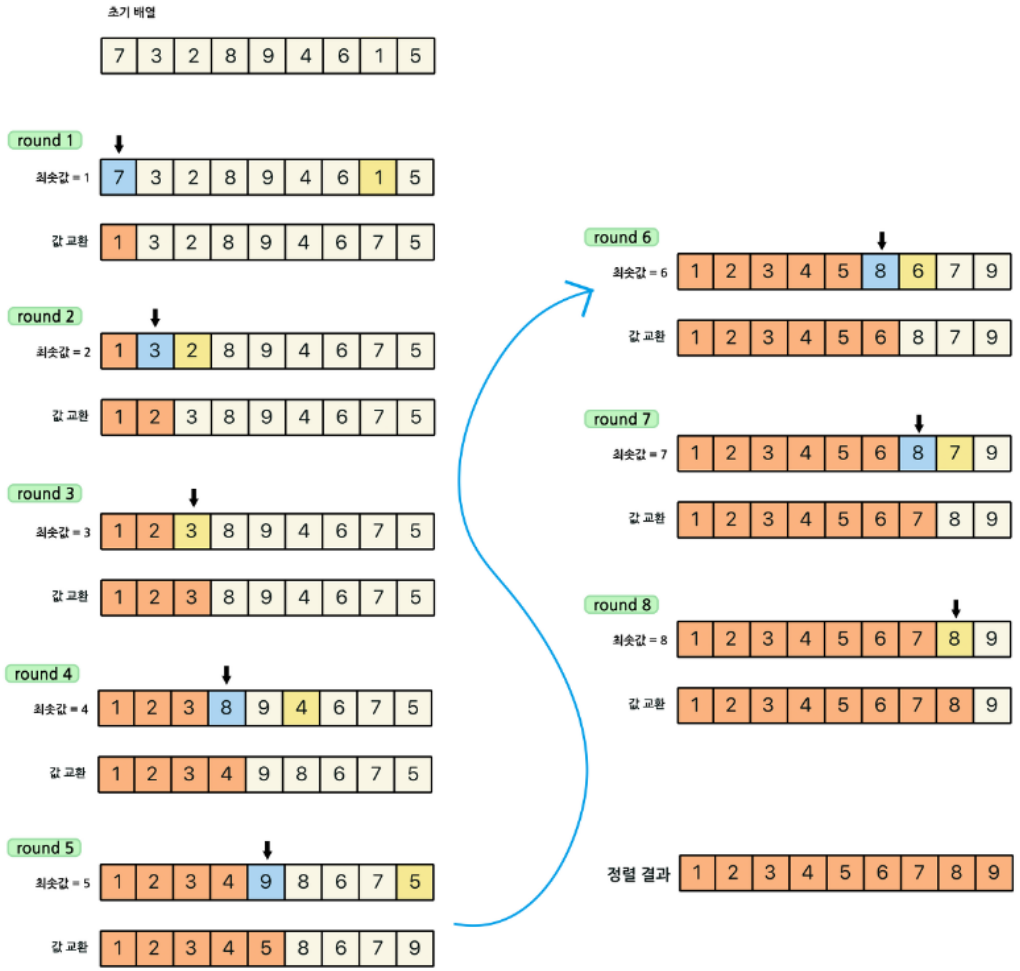
※스택 오버플로우: 변수의 크기가 Stack 보다 크거나, 함수를 무한으로 호출하고 있을 때, 혹은 Stack을 넘어가 다른 곳에 위치하는 경우 발생

Unit 03 | 정렬 알고리즘 소개

19기 정규세션
TOBIG'S 18기 지윤혁

선택정렬

정렬이 되지 않은 숫자들 중에서 최소값을 선택하여
배열의 첫번째 요소와 교환하는 정렬알고리즘



선택정렬

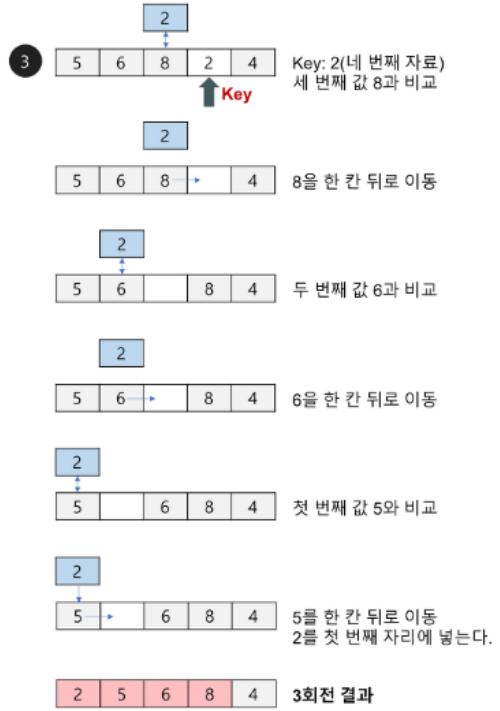
정렬이 되지 않은 숫자들 중에서 최소값을 선택하여
배열의 첫번째 요소와 교환하는 정렬알고리즘

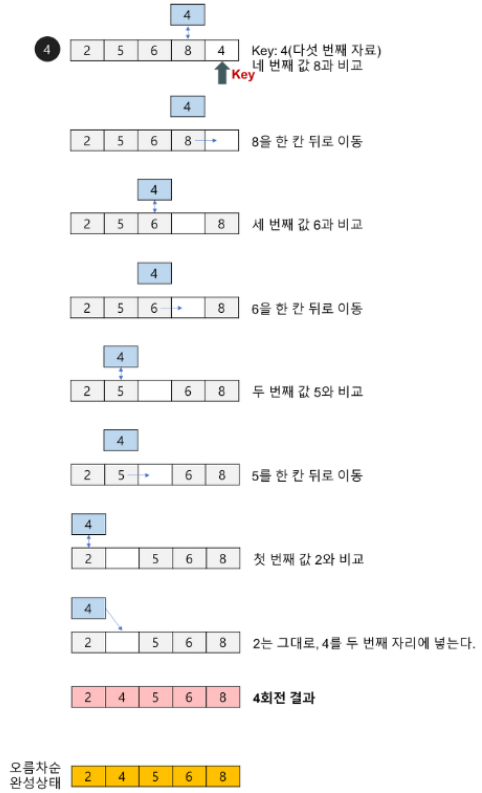
```
def selection_sort(arr):
  for i in range(len(arr)-1):
    min_idx = i
    for j in range(i+1, len(arr)):
      if arr[min_idx] > arr[j]:
        mim_idx = j

    arr[i], arr[min_idx] = arr[min_idx], arr[i]

  return arr
```

19기 정규세션
TOBIG'S 18기 지윤혁

삽입정렬

두번째 원소부터 시작해 그 앞에 존재하는 원소들과 비교하여
삽입할 위치를 찾아 삽입하는 정렬 알고리즘





19기 정규세션
TOBIG'S 18기 지윤혁

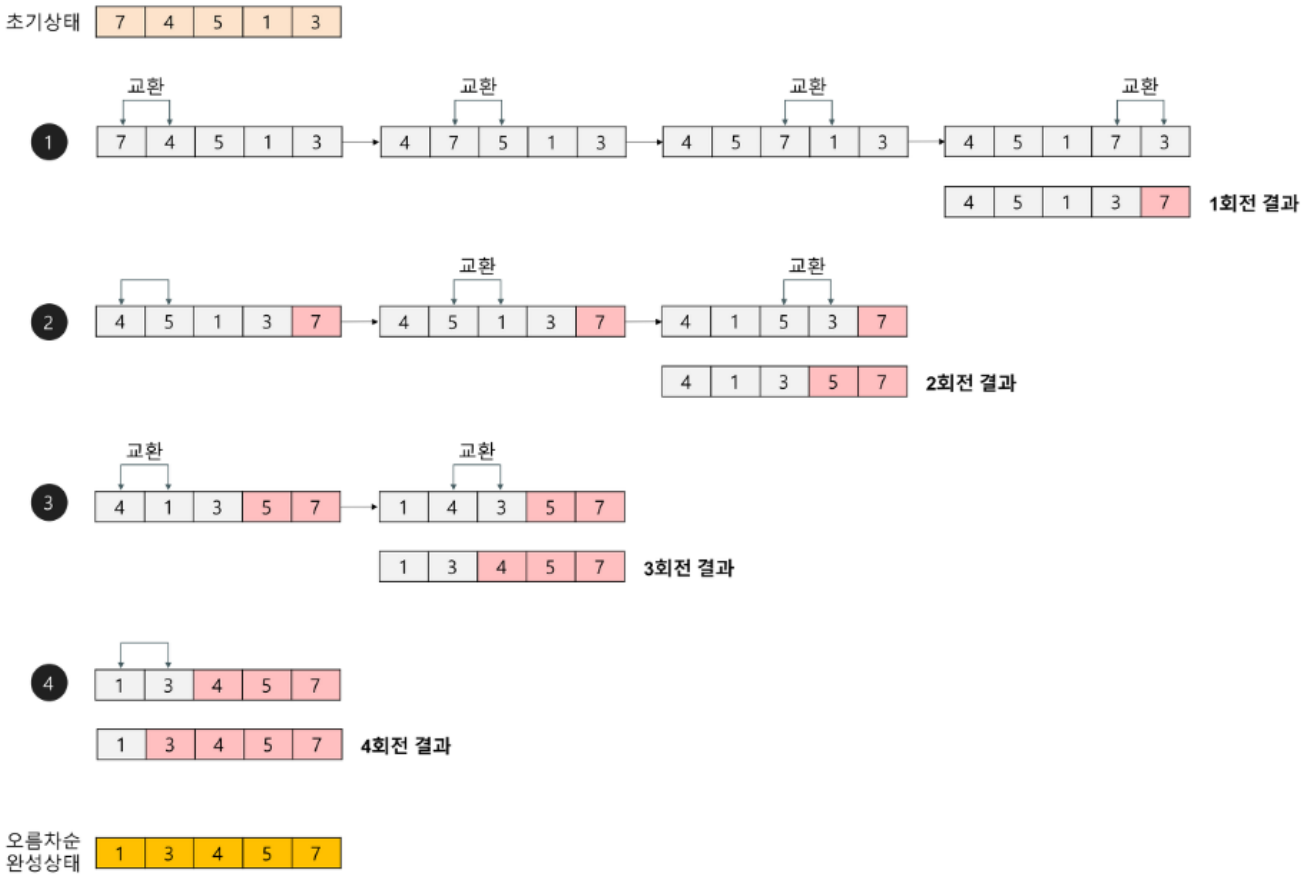
삽입정렬

두번째 원소부터 시작해 그 앞에 존재하는 원소들과 비교하여
삽입할 위치를 찾아 삽입하는 정렬 알고리즘

```
def insertion_sort(arr):
  for i in range(1, len(arr)):
    key_item = arr[i]
    j = i-1

    while j >= 0 and arr[j] > key_item:
      arr[j+1] = arr[j]
      j -= 1

    arr[j+1] = key_item

  return arr
```

19기 정규세션
TOBIG'S 18기 지윤혁

버블정렬 서로 인접한 두 원소를 비교하여 정렬하는 알고리즘



19기 정규세션
TOBIG'S 18기 지윤혁

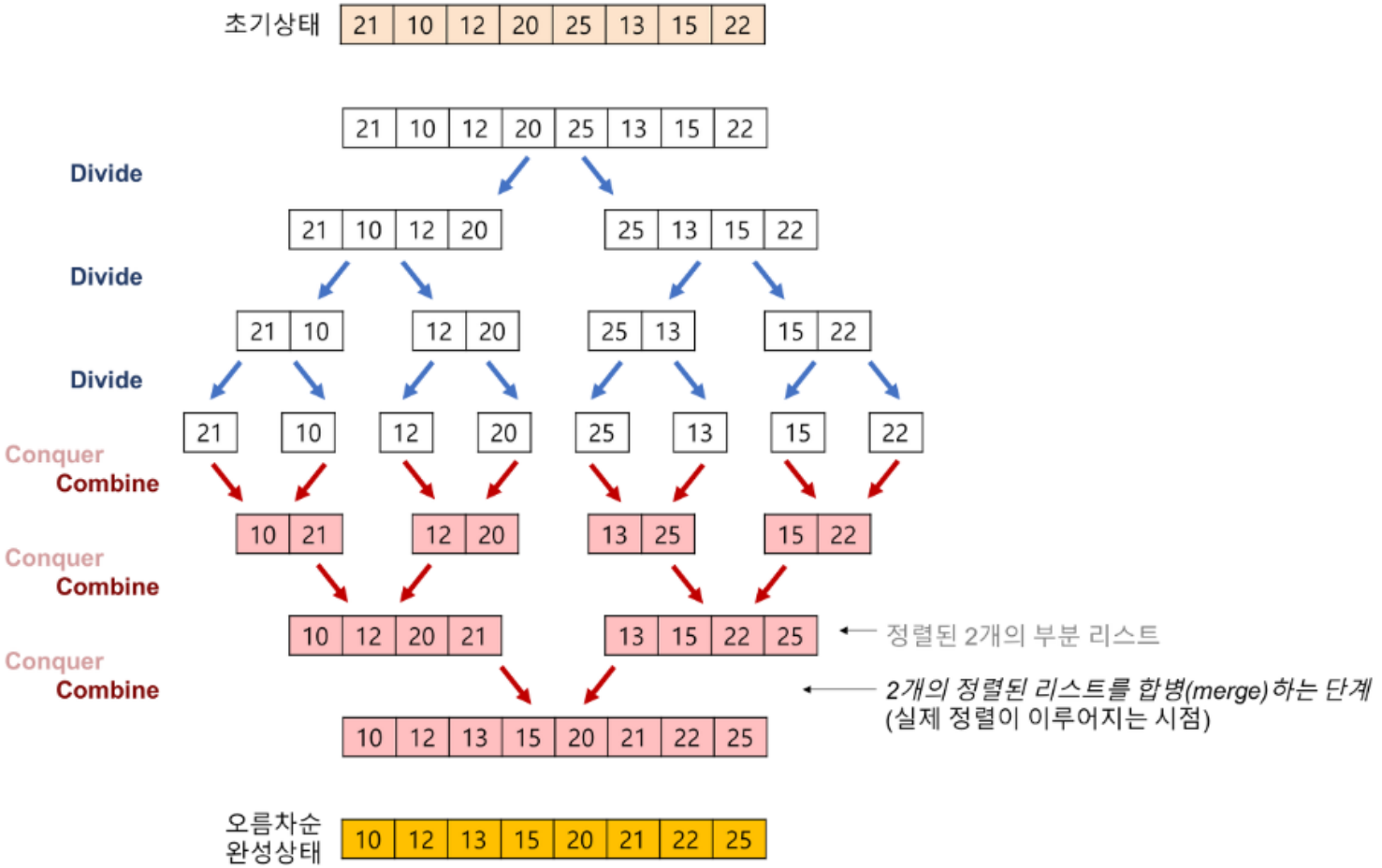
버블정렬 서로 인접한 두 원소를 비교하여 정렬하는 알고리즘

```python
def bubble_sort(arr):
  for i in range(len(arr)-1):
    done_sort = True

    for j in range(len(arr)-i-1):
      if arr[j] > arr[j+1]:
        arr[j], arr[j+1] = arr[j+1], arr[j]

        done_sort = True

    if done_sort:
      break

  return arr
```

19기 정규세션
TOBIG'S 18기 지윤혁

Merge Sort

주어진 배열을 크기가 1인 배열로 분할하고 합병하면서
정렬을 진행하는 분할정복 알고리즘



19기 정규세션
TOBIG'S 18기 지윤혁

Merge Sort

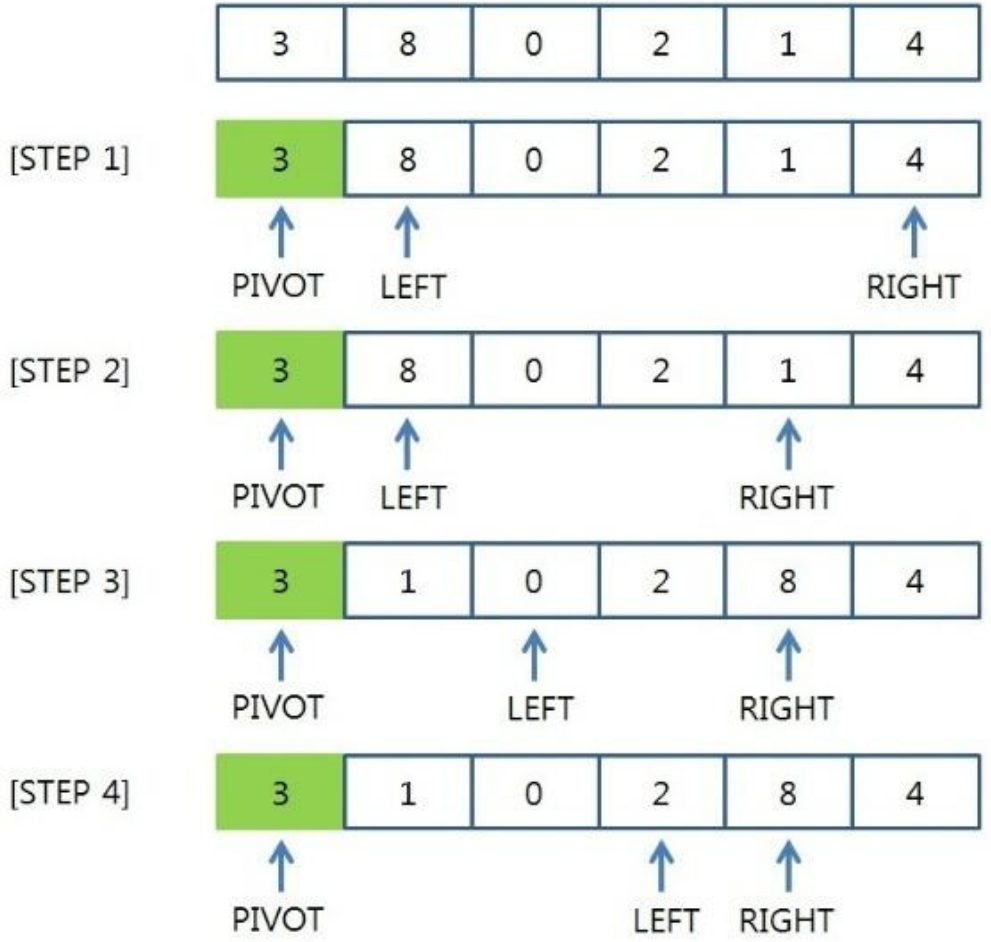
주어진 배열을 크기가 1인 배열로 분할하고 합병하면서
정렬을 진행하는 분할정복 알고리즘

```python
def merge(left, right):
  left_len = len(left)
  right_len = len(right)

  result = []
  left_index = right_index = 0

  while len(result) < left_len + right_len:
    if left[left_index] <= right[right_index]:
      result.append(left[left_index])
      left_index += 1
    else:
      result.append(right[right_index])
      right_index += 1

    if right_index == right_len:
      result.extend(left[left_index:])
      break
    if left_index == left_len:
      result.extend(right[right_index:])
      break

  return result

def merge_sort(arr):
  if len(arr) < 2:
    return arr

  mid_index = len(arr) // 2
  left = merge_sort(arr[:mid_index])
  right = merge_sort(arr[mid_index:])

  return merge(left, right)
```

19기 정규세션
TOBIG'S 18기 지윤혁

Quick Sort

Merge Sort와 다르게 리스트를 비균등하게 분할
Pivot을 설정하고 Pivot보다 큰값, 작은값으로 분할하여 정렬



19기 정규세션
TOBIG'S 18기 지윤혁

Quick Sort

Merge Sort와 다르게 리스트를 비균등하게 분할
Pivot을 설정하고 Pivot보다 큰값, 작은값으로 분할하여 정렬

```python
from random import randint

def quicksort(arr):
  if len(arr) < 2:
    return arr

  low, same, high = [], [], []
  pivot = arr[randint(0, len(arr)-1]

  for item in arr:
    if item < pivot:
      low.append(item)
    elif item == pivot:
      same.append(item)
    elif item > pivot:
      high.append(item)

  return quicksort(low)+same+quicksort(high)
```

Unit 04 | 7주차 과제 소개

19기 정규세션
TOBIG'S 18기 지윤혁

문제 1. 222-풀링

문제

1. 행렬을 2×2 정사각형으로 나눈다.

-1	2	14	7	4	-5	8	9
10	6	23	2	-1	-1	7	11
9	3	5	-2	4	4	6	6
7	15	0	8	21	20	6	6
19	8	12	-8	4	5	2	9
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24

2. 각 정사각형에서 2번째로 큰 수만 남긴다. 여기서 2번째로 큰 수란, 정사각형의 네 원소를 크기순으로 $a_4 \le a_3 \le a_2 \le a_1$ 라 했을 때, 원소 a_2를 뜻한다.

-1	2	**14**	7	4	-5	8	**9**
10	**6**	23	2	**-1**	-1	7	11
9	3	**5**	-2	4	4	**6**	6
7	15	0	8	21	**20**	6	6
19	**8**	12	-8	4	**5**	2	9
1	2	3	**4**	5	6	7	**8**
9	10	11	12	13	14	15	16
17	18	**19**	20	**21**	22	**23**	24

6	14	-1	9
9	5	20	6
8	4	5	8
17	19	21	23

3. 2번 과정에 의해 행렬의 크기가 줄어들게 된다.

입력

첫째 줄에 $N(2 \le N \le 1024)$이 주어진다. N은 항상 2의 거듭제곱 꼴이다. ($N=2^K$, $1 \le K \le 10$)

다음 N개의 줄마다 각 행의 원소 N개가 차례대로 주어진다. 행렬의 모든 성분은 -10,000 이상 10,000 이하의 정수이다.

출력

마지막에 남은 수를 출력한다.

예제 입력 1 복사

```
4
-6 -8 7 -4
-5 -5 14 11
11 11 -1 -1
4 9 -2 -4
```

예제 출력 1 복사

```
11
```

예제 입력 2 복사

```
8
-1 2 14 7 4 -5 8 9
10 6 23 2 -1 -1 7 11
9 3 5 -2 4 4 6 6
7 15 0 8 21 20 6 6
19 8 12 -8 4 5 2 9
1 2 3 4 5 6 7 8
9 10 11 12 13 14 15 16
17 18 19 20 21 22 23 24
```

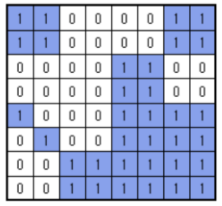
예제 출력 2 복사

```
17
```

문제 2. 색종이 만들기

문제

아래 <그림 1>과 같이 여러개의 정사각형칸들로 이루어진 정사각형 모양의 종이가 주어져 있고, 각 정사각형들은 하얀색으로 칠해져 있거나 파란색으로 칠해져 있다. 주어진 종이를 일정한 규칙에 따라 잘라서 다양한 크기를 가진 정사각형 모양의 하얀색 또는 파란색 색종이를 만들려고 한다.

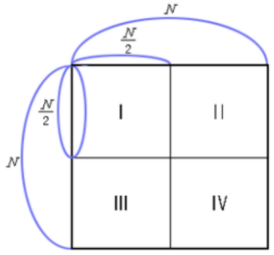


<그림 1> 8×8 종이

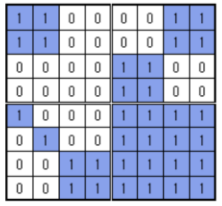
전체 종이의 크기가 N×N(N=2^k, k는 1 이상 7 이하의 자연수) 이라면 종이를 자르는 규칙은 다음과 같다.

전체 종이가 모두 같은 색으로 칠해져 있지 않으면 가로와 세로로 중간 부분을 잘라서 <그림 2>의 I, II, III, IV와 같이 똑같은 크기의 네 개의 N/2 × N/2색종이로 나눈다. 나누어진 종이 I, II, III, IV 각각에 대해서도 앞에서와 마찬가지로 모두 같은 색으로 칠해져 있지 않으면 같은 방법으로 똑같은 크기의 네 개의 색종이로 나눈다. 이와 같은 과정을 잘라진 종이가 모두 하얀색 또는 모두 파란색으로 칠해져 있거나, 하나의 정사각형 칸이 되어 더 이상 자를 수 없을 때까지 반복한다.

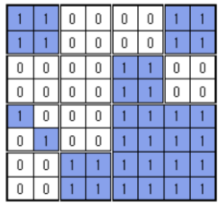
위와 같은 규칙에 따라 잘랐을 때 <그림 3>은 <그림 1>의 종이를 처음 나눈 후의 상태를, <그림 4>는 두 번째 나눈 후의 상태를, <그림 5>는 최종적으로 만들어진 다양한 크기의 9장의 하얀색 색종이와 7장의 파란색 색종이를 보여주고 있다.



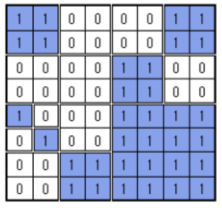
<그림 2>



<그림 3> 처음 나눈 후의 상태



<그림 4> 두 번째 나눈 후의 상태



<그림 5> 최종적으로 나누어진 색종이들

입력으로 주어진 종이의 한 변의 길이 N과 각 정사각형칸의 색(하얀색 또는 파란색)이 주어질 때 잘라진 하얀색 색종이와 파란색 색종이의 개수를 구하는 프로그램을 작성하시오.

입력

첫째 줄에는 전체 종이의 한 변의 길이 N이 주어져 있다. N은 2, 4, 8, 16, 32, 64, 128 중 하나이다. 색종이의 각 가로줄의 정사각형칸들의 색이 윗줄부터 차례로 둘째 줄부터 마지막 줄까지 주어진다. 하얀색으로 칠해진 칸은 0, 파란색으로 칠해진 칸은 1로 주어지며, 각 숫자 사이에는 빈칸이 하나씩 있다.

출력

첫째 줄에는 잘라진 하얀색 색종이의 개수를 출력하고, 둘째 줄에는 파란색 색종이의 개수를 출력한다.

예제 입력 1 복사

```
8
1 1 0 0 0 0 1 1
1 1 0 0 0 0 1 1
0 0 0 0 1 1 0 0
0 0 0 0 1 1 0 0
1 0 0 0 1 1 1 1
0 1 0 0 1 1 1 1
0 0 1 1 1 1 1 1
0 0 1 1 1 1 1 1
```

예제 출력 1 복사

```
9
7
```

19기 정규세션
TOBIG'S 18기 지윤혁

문제 3. 종이의 개수

문제

N×N크기의 행렬로 표현되는 종이가 있다. 종이의 각 칸에는 -1, 0, 1 중 하나가 저장되어 있다. 우리는 이 행렬을 다음과 같은 규칙에 따라 적절한 크기로 자르려고 한다.

1. 만약 종이가 모두 같은 수로 되어 있다면 이 종이를 그대로 사용한다.
2. (1)이 아닌 경우에는 종이를 같은 크기의 종이 9개로 자르고, 각각의 잘린 종이에 대해서 (1)의 과정을 반복한다.

이와 같이 종이를 잘랐을 때, -1로만 채워진 종이의 개수, 0으로만 채워진 종이의 개수, 1로만 채워진 종이의 개수를 구해내는 프로그램을 작성하시오.

입력

첫째 줄에 N($1 \le N \le 3^7$, N은 3^k 꼴)이 주어진다. 다음 N개의 줄에는 N개의 정수로 행렬이 주어진다.

출력

첫째 줄에 -1로만 채워진 종이의 개수를, 둘째 줄에 0으로만 채워진 종이의 개수를, 셋째 줄에 1로만 채워진 종이의 개수를 출력한다.

예제 입력 1 복사

```
9
0 0 0 1 1 1 -1 -1 -1
0 0 0 1 1 1 -1 -1 -1
0 0 0 1 1 1 -1 -1 -1
1 1 1 0 0 0 0 0 0
1 1 1 0 0 0 0 0 0
1 1 1 0 0 0 0 0 0
0 1 -1 0 1 -1 0 1 -1
0 -1 1 0 1 -1 0 1 -1
0 1 -1 1 0 -1 0 1 -1
```

예제 출력 1 복사

```
10
12
11
```

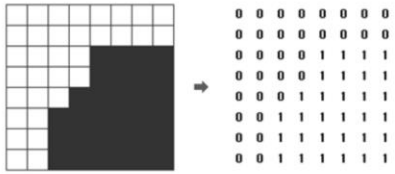
19기 정규세션
TOBIG'S 18기 지윤혁

문제 4. 쿼드트리

문제

흑백 영상을 압축하여 표현하는 데이터 구조로 쿼드 트리(Quad Tree)라는 방법이 있다. 흰 점을 나타내는 0과 검은 점을 나타내는 1로만 이루어진 영상(2차원 배열)에서 같은 숫자의 점들이 한 곳에 많이 몰려있으면, 쿼드 트리에서는 이를 압축하여 간단히 표현할 수 있다.

주어진 영상이 모두 0으로만 되어 있으면 압축 결과는 "0"이 되고, 모두 1로만 되어 있으면 압축 결과는 "1"이 된다. 만약 0과 1이 섞여 있으면 전체를 한 번에 나타내지를 못하고, 왼쪽 위, 오른쪽 위, 왼쪽 아래, 오른쪽 아래, 이렇게 4개의 영상으로 나누어 압축하게 되며, 이 4개의 영역을 압축한 결과를 차례대로 괄호 안에 묶어서 표현한다



위 그림에서 왼쪽의 영상은 오른쪽의 배열과 같이 숫자로 주어지며, 이 영상을 쿼드 트리 구조를 이용하여 압축하면 "`(0(0011)(0(0111)01)1)`"로 표현된다. N ×N 크기의 영상이 주어질 때, 이 영상을 압축한 결과를 출력하는 프로그램을 작성하시오.

입력

첫째 줄에는 영상의 크기를 나타내는 숫자 N 이 주어진다. N 은 언제나 2의 제곱수로 주어지며, 1 ≤ N ≤ 64의 범위를 가진다. 두 번째 줄부터는 길이 N의 문자열이 N개 들어온다. 각 문자열은 0 또는 1의 숫자로 이루어져 있으며, 영상의 각 점들을 나타낸다.

출력

영상을 압축한 결과를 출력한다.

예제 입력 1 복사

```
8
11110000
11110000
00011100
00011100
11110000
11110000
11110011
11110011
```

예제 출력 1 복사

```
((110(0101))(0010)1(0001))
```

7주차 알고리즘 과제

백준 17829번 222-풀링	https://www.acmicpc.net/problem/17829
백준 2630번 색종이 만들기	https://www.acmicpc.net/problem/2630
백준 1780번 종이의 개수	https://www.acmicpc.net/problem/1780
백준 1992번 쿼드트리	https://www.acmicpc.net/problem/1992

7주차 알고리즘 과제 기한 : 3월 14일 23:59

감사합니다