분할 정복 (Divide and Conquer)

01 분할 정복 이론

02 분할 정복 예시 문제 풀이

신 제 용

ZEROBASE SCHOOL

01 분할 정복 이론

큰 문제를 작은 부분 문제로 나누어 해결하는 분할 정복을 학습합니다.

학습 키워드 – 분할 정복, Divide and conquer Top-Down

Chapter 01
분할 정복 이론

ZEROBASE SCHOOL

분할 정복 (Divide and conquer)

- 큰 문제를 작은 부분 문제로 나누어 해결하는 방법
  - 합병 정렬, 퀵 정렬, 이진 검색, …

- 분할 정복 과정
  1. 문제를 하나 이상의 부분 문제(sub-problem)로 분할
  2. 부분 문제를 각각 해결
  3. 부분 문제의 해답을 통합하여 원래 문제를 해결

분할 정복의 장/단점

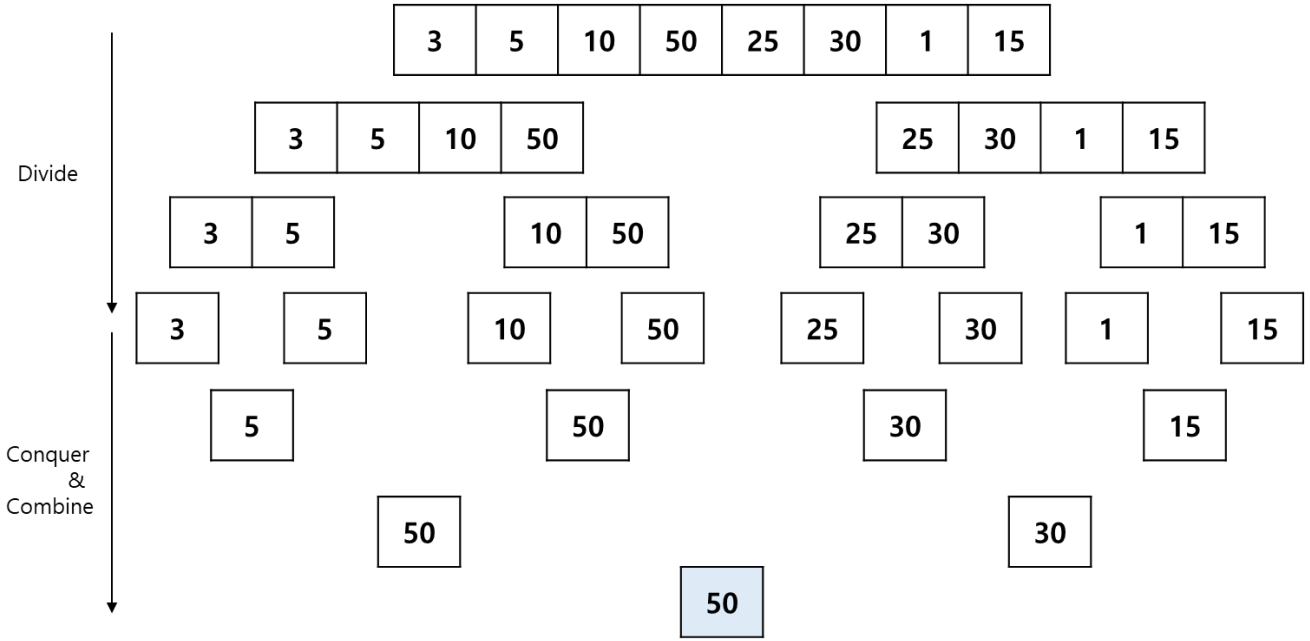
- 장점
  - 문제를 나누어 처리하며 어려운 문제 해결 가능
  - 병렬 처리에 적합한 계산 구조

- 단점
  - 메모리를 많이 사용 (재귀 호출 구조)

ZEROBASE SCHOOL

분할 정복 예시 (1)

- 최대값 찾기



ZEROBASE SCHOOL

분할 정복 예시 (2)

- 최대값 찾기

3	5	10	50	25	30	1	15

```python
def get_max(arr, left, right):
    if left == right:
        return arr[left]

    mid = (left + right) // 2
    left = get_max(arr, left, mid)
    right = get_max(arr, mid + 1, right)

    return left if left > right else right
```


ZEROBASE SCHOOL

02 분할 정복 예시 문제 풀이

다음 챕터에서는 분할 정복을 활용하는
예시 문제를 풀이하는 방법을 학습합니다.

Chapter 01
분할 정복 이론

ZEROBASE SCHOOL

02 분할 정복 예시 문제 풀이

분할 정복을 활용하는 대표적인 예시 문제를 학습합니다.

학습 키워드 – 분할 정복, 구현

Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL

Problem1

문제 설명

흰색(0)과 검은색(1)로만 이루어진 이진영상(binary image)은 다양한 방식으로 압축할 수 있다. 여러 압축 방법 중에, 당신은 쿼드트리(quad-tree) 방식으로 압축하고자 한다.

쿼드트리 압축 방식은 아래와 같다.

- 압축하려는 영상 전체가 0이면 `"0"`, 전체가 1이면 `"1"`로 압축한다.
- 영상 전체가 같은 값이 아니라면, 영상을 동일한 크기로 4분할하여 부분 영상 별로 압축한다.
  - 각 부분 영상을 압축하는 방법은 전체 영상을 압축하는 방법과 같다.
  - 각각의 압축 결과를 `"(좌상단 우상단 좌하단 우하단)"` 과 같이 출력한다. (각 문자 사이에 공백은 쓰지 않는다. ex) (0110))

아래와 같은 영상이 있다고 예를 들어보자.

```
0, 0, 0, 0, 1, 1, 1, 1
0, 0, 0, 0, 1, 1, 0, 0
0, 0, 0, 0, 0, 0, 1, 1
0, 0, 0, 0, 0, 0, 1, 1
1, 1, 0, 0, 0, 0, 0, 0
1, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 1
```

이 때, 압축 결과는 아래와 같다.

`(0(1(1100)01)((1110)000)(000(0001)))`

위 쿼드트리 압축 결과를 출력하는 프로그램을 작성하시오.

Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL

입출력 예시

예시1

- 입력

```
0, 0
0, 0
```

- 출력

```
0
```

- 해설
  - 영상 전체가 0이기 때문에, 0으로 압축할 수 있다.

예시2

- 입력

```
0, 1
1, 0
```

- 출력

```
(0110)
```

- 해설
  - 영상 전체가 같은 수가 아니므로, `좌상단, 우상단, 좌하단, 우하단` 순으로 숫자를 나열한다. 압축을 했기 때문에 양 옆에 괄호를 필요로 한다.

Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL

예시3

- 입력

```
0, 0, 0, 0, 1, 1, 1, 1
0, 0, 0, 0, 1, 1, 1, 1
0, 0, 0, 0, 1, 1, 1, 1
0, 0, 0, 0, 1, 1, 1, 1
1, 1, 1, 1, 0, 0, 1, 1
1, 1, 1, 1, 0, 0, 1, 1
1, 1, 1, 1, 1, 1, 0, 0
1, 1, 1, 1, 1, 1, 0, 0
```

- 출력

```
(011(0110))
```

- 해설
  - 영상 전체가 같은 수가 아니므로, 좌상단, 우상단, 좌하단, 우하단 순으로 인코딩해야 한다.
    - 좌상단은 전체가 `0`이므로, `0`으로 압축된다.
    - 우상단은 전체가 `1`이므로, `1`로 압축된다.
    - 좌하단은 전체가 `1`이므로, `1`로 압축된다.
    - 우하단은 전체가 같은 수가 아니므로, 좌상단, 우상단, 좌하단, 우하단 순으로 인코딩해야 한다.
      - 좌상단은 전체가 `0`이므로, `0`으로 압축된다.
      - 우상단은 전체가 `1`이므로, `1`로 압축된다.
      - 좌하단은 전체가 `1`이므로, `1`로 압축된다.
      - 우하단은 전체가 `0`이므로, `0`으로 압축된다.
      - 따라서 이 영역은 `(0110)`으로 압축된다.
  - 위 결과에 따라, 전체 영역은 `(011(0110))`으로 압축된다.

ZEROBASE SCHOOL

Problem2

문제 설명

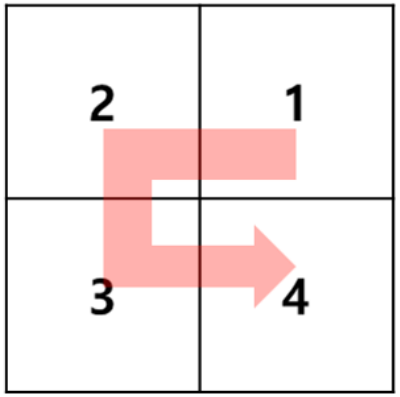
담디는 디귿(ㄷ)자 모양이 너무나도 좋아서, 2차원 배열에 숫자를 디귿자로 배치하려고 한다.

깔끔한 것을 좋아하는 담디는 정방형(높이와 너비가 같은) 2차원 배열만을 사용한다.

숫자를 채우는 구체적인 방법은 아래와 같다.

- 배열의 높이와 너비는 `n`으로 같으며, 항상 2의 제곱수이다.
- 배열을 높이와 너비를 반반씩 나누어 4개의 부분 배열을 만든다.
  - 우측 상단, 좌측 상단, 좌측 하단, 우측 하단의 순서로 숫자를 채운다. (ㄷ자를 한 붓 그리기 한 순서)
  - 각 부분 배열을 채우는 방법은 이 방법을 재귀적으로 사용하여 숫자를 채운다.

즉, 예를 들면 `n=2`인 경우에는 아래와 같이 디귿자 모양으로 곧바로 숫자를 채운다.

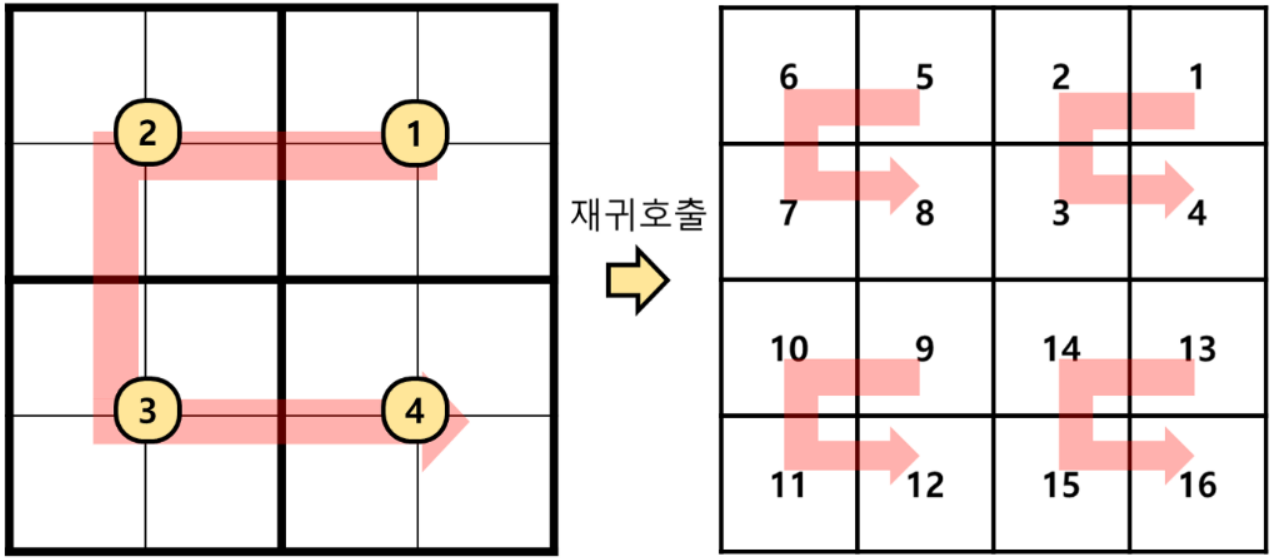


Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL

`n=4` 인 경우에는 아래와 같이 재귀적으로 순서를 정해서 숫자를 채운다.



이 때, `i` 행 `j` 열에 위치한 값을 반환하는 프로그램을 작성하시오. (행과 열은 `0` 부터 시작해서 `n-1` 까지의 값을 가진다.)

매개변수 형식

`n = 4` `i = 1` `j = 3`

반환값 형식

`4`

예시 입출력 설명

위의 `n=4` 인 경우에서 `(1, 3)` 위치의 값인 `4` 를 반환

Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL

Problem3

문제 설명

정수가 담긴 리스트 `nums` 가 주어졌다.

연속된 부분 배열의 합 중 가장 큰 값을 출력하세요.

매개변수 형식

```
nums = [-5, 0, -3, 4, -1, 3, 1, -5, 8]
```

반환값 형식

```
10
```

Chapter 02

분할 정복 예시 문제 풀이

ZEROBASE SCHOOL