씨앗

senior

Week 1

Date : 23.03.?

시작하기에 앞서

1. 씨앗 활동은 강의가 아니라 스터디입니다!
   - 제가 잘 알지 못하는 부분이 있을 수 있습니다.
     함께 공부하는 시간이 되었으면 좋겠습니다.

2. 활동은 이론15분 + 문제 풀이60분 + α 입니다.
   - 질문이 있으시면 언제든지:)
   - 문제풀이 or 정답코드를 원하시면 제공해드립니다.

오늘의 내용

1. DP, 동적 계획법이란
2. 예제 풀어보기
3. 문제 풀이

Dynamic Programming

문제의 답을 얻기 위하여 작은 부분 문제들을 해결하고
해결한 문제들의 답을 사용하여 최종 답을 얻는 것

메모지에이션(memoziation) :
계산결과를 메모리에 저장하여 동일한 계산의 반복을 막는 것

중요 : 점화식 or 값들의 관계를 빠르게 파악!

언제 사용할까

1. 문제의 결과를 구하기 위해 작은 문제들의 답을 사용한다.
2. 작은 문제의 계산이 반복된다.

어떻게 사용할까

1. 문제의 최적해를 어떻게 구할 지 생각한다.
   => 최적해는 어떻게 구할 까 => 점화식 도출

2. 반복문 or 재귀함수를 통해 작은 문제들의 계산을 구현한다.

3. 적절한 자료구조에 계산결과를 저장해가며 큰 값에 접근한다.
   (배열, 벡터, 덱등)

왜 중요한가?

- 가장 다양하면서도 많은 문제가 있음.
- 정말 쉬운 난이도부터 정말 어려운 난이도까지 고루 분포.
- 다양한 알고리즘과 혼용(비트 마스킹, 이분탐색, 구간합 등)
- 익히고 나면, 정말 많은 문제를 풀 수 있게 됨.

Ex) 피보나치 함수

시간 복잡도 : $O(2^n) \rightarrow O(n)$ 으로 개선

SEED > Senior > week1 > fibo_recursion.py > ...

```python
def fib_recursion(n):
    if n == 0 : return 0
    elif n == 1 : return 1
    else :
        return fib_recursion(n - 1) + fib_recursion(n - 2)

N = int(input())
print(fib_recursion(N))
```

SEED > Senior > week1 > fibo_DP.py > ...

```python
N = int(input())
# 1. 배열을 하나 만들기
DP = [False] * (N + 1)
# 2. 초기값 넣기
DP[0] = 0
DP[1] = 1
# 3. 답을 구할 때까지 반복!
for i in range(2,N + 1):
    if DP[i] : continue
    else : DP[i] = DP[i - 1] + DP[i - 2]
# 4. 결과출력
print(DP[N])
```

구현 방식

bottom-up

- 작은 문제에서 큰 문제로 진행
- 함수 호출 횟수가 적음
- 시간과 메모리, 조금이나마 절약

top-down

- 큰 값에서 작은 값으로 진행
- 가독성이 좋음
- 점화식을 이해하기 쉬움

문제 형태에 따라 적합한 방식을 선택

오늘의 문제

Normal

1003번 : 피보나치 함수

https://www.acmicpc.net/problem/1003

10844번 : 쉬운 계단 수

https://www.acmicpc.net/problem/10844

9461번 : 파도반 수열

https://www.acmicpc.net/problem/9461

Advanced

2011번 : 암호코드

https://www.acmicpc.net/problem/2011

2225번 : 합분해

https://www.acmicpc.net/problem/2225

Harder

2629번 : 양팔저울

https://www.acmicpc.net/problem/2629

1256번 : 사전

https://www.acmicpc.net/problem/1256

그 밖

이친수(pinary number) 시리즈

- 2193번 : 이친수
- 2201번 : 이친수 찾기

최장 증가 부분 수열(Longest Increasing Subsequence)

- 11053번 : 가장 긴 증가하는 부분 수열
- 11054번 : 가장 긴 바이토닉 부분 수열
- 14002번 : 가장 긴 증가하는 부분 수열4

배낭 채우기(Knapsack Problem)

- Fractional Knapsack (자를 수 있는 보석으로 채우기)
- 0-1 Knapsack (자를 수 없는 보석으로 채우기)