2주차 주간학습보고서(3/29~4/4)

분류 1	전공		
팀 명	정융팡팡	제출자	최수지
학습 일시	2021.03.29 17:00 ~ 2021.03.29 19:00	마감 일시	2021.03.29 19:00
제출 일시	2021.04.01 00:12	수정 일시	2021.04.02 17:28

[2021-1 광운스터디그룹]

주간학습보고서

학습주제	동적 프로그래밍
학습장소	기념관 2층 로비
학습방법 및 학습개요	동적 프로그래밍 1. 동적 프로그래밍의 개념 2. 문제(백준 알고리즘 : https://www.acmicpc.net/) 1) 2839_설탕배달 2) 2748_피보나치 수2 3) 1912_연속합 4) 11726_2*n 타일링 5) 2098_외판원 순회 6) 1520_내리막 길 3. 정리 깃허브 인증
다음 주 계획	**주제 : 정렬** **일시 : 2021.04.05** **장소 : 기념관 2층** **보고서 제출자 : 안희승**

* 학습개요의 내용을 구체적으로 작성해 주세요.
<2주차 학습내용>
1. 동적 프로그래밍의 개념
Dynamic Programming(동적 프로그래밍)은 문제를 한번에 해결하기 어려울 때 소문제로 나누어 푸는 기법이다. 이 때, 소문제들을 풀다보면 같은 문제들을 반복하여 푸는 경우가 생긴다. 이 경우에 매번 그 값을 다시 계산하지 않고 저장해두었다가 재사용하는 방법이다.
2. 문제
1) 2839번 설탕 배달

설탕 배달 성공 출처 다국어 분류

☆ 한국어

브론즈 I

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	156401	49671	39207	32.896%

문제

상근이는 요즘 설탕공장에서 설탕을 배달하고 있다. 상근이는 지금 사탕가게에 설탕을 정확하게 N킬로그램을 배달해야 한다. 설탕공장에서 만드는 설탕은 봉지에 담겨져 있다. 봉지는 3킬로그램 봉지와 5킬로그램 봉지가 있다.

상근이는 귀찮기 때문에, 최대한 적은 봉지를 들고 가려고 한다. 예를 들어, 18킬로그램 설탕을 배달해야 할 때, 3킬로그램 봉지 6개를 가져가도 되지만, 5킬로그램 3개와 3킬로그램 1개를 배달하면, 더 적은 개수의 봉지를 배달할 수 있다.

상근이가 설탕을 정확하게 N킬로그램 배달해야 할 때, 봉지 몇 개를 가져가면 되는지 그 수를 구하는 프로그램을 작성하시오.

-풀이 (최수지)

```
N = int(input())
result = 0

while N >= 0:
    if N % 5 == 0:
        result += (N // 5)
        print(result)
        break
    N -= 3
    result += 1
else:
    print(-1)
```

2) 2748번 피보나치 수2

피보나치 수 2 성공 다국어 분류

☆ 한국어

브론즈 I

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	54913	21249	17710	38.750%

문제

피보나치 수는 0과 1로 시작한다. 0번째 피보나치 수는 0이고, 1번째 피보나치 수는 1이다. 그 다음 2번째 부터는 바로 앞 두 피보나치 수의 합이 된다.

이를 식으로 써보면 $F_n = F_{n-1} + F_{n-2}$ ($n \geq 2$)가 된다.

n=17일때 까지 피보나치 수를 써보면 다음과 같다.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597

n이 주어졌을 때, n번째 피보나치 수를 구하는 프로그램을 작성하시오.

-풀이 (최수지)

```
n = int(input())
a,b = 0,1
li = []
for x in range(n+1):
    li.append(a)
    a,b = b, a+b

print(li[n])
```

3) 1912번 연속합

연속합 성공 분류

☆

실버 II

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초 (추가 시간 없음)	128 MB	80282	25077	17342	30.376%

문제

n개의 정수로 이루어진 임의의 수열이 주어진다. 우리는 이 중 연속된 몇 개의 수를 선택해서 구할 수 있는 합 중 가장 큰 합을 구하려고 한다. 단, 수는 한 개 이상 선택해야 한다.

예를 들어서 10, -4, 3, 1, 5, 6, -35, 12, 21, -1 이라는 수열이 주어졌다고 하자. 여기서 정답은 12+21인 33이 정답이 된다.

-풀이 (최수지)

```
n = int(input())
li = [int(x) for x in input().strip().split()]
result = [li[0]]

for x in range(n-1):
    result.append(max(result[x] + li[x+1],li[x+1]))

print(max(result))
```

4) 11726번 2*n 타일링

2×n 타일링 성공 분류

☆

실버 III

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	256 MB	78613	29210	21326	34.902%

문제

2×n 크기의 직사각형을 1×2, 2×1 타일로 채우는 방법의 수를 구하는 프로그램을 작성하시오.

아래 그림은 2×5 크기의 직사각형을 채운 한 가지 방법의 예이다.

-풀이 (최수지)

```
import sys

n = int(sys.stdin.readline())
a,b = 0, 1
result = 0
li = []

for _ in range(n+2):
    li.append(result+a)
    a,b = b, a+b

print(li[n+1]%10007)
```

5) 2098번 외판원 순회

외판원 순회 성공 분류

☆

골드 I

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	22736	6098	3634	28.605%

문제

외판원 순회 문제는 영어로 Traveling Salesman problem (TSP) 라고 불리는 문제로 computer science 분야에서 가장 중요하게 취급되는 문제 중 하나이다. 여러 가지 변종 문제가 있으나, 여기서는 가장 일반적인 형태의 문제를 살펴보자.

1번부터 N번까지 번호가 매겨져 있는 도시들이 있고, 도시들 사이에는 길이 있다. (길이 없을 수도 있다) 이제 한 외판원이 어느 한 도시에서 출발해 N개의 도시를 모두 거쳐 다시 원래의 도시로 돌아오는 순회 여행 경로를 계획하려고 한다. 단, 한 번 갔던 도시로는 다시 갈 수 없다. (맨 마지막에 여행을 출발했던 도시로 돌아오는 것은 예외) 이런 여행 경로는 여러 가지가 있을 수 있는데, 가장 적은 비용을 들이는 여행 계획을 세우고자 한다.

각 도시간에 이동하는데 드는 비용은 행렬 W[i][j]형태로 주어진다. W[i][j]는 도시 i에서 도시 j로 가기 위한 비용을 나타낸다. 비용은 대칭적이지 않다. 즉, W[i][j] 는 W[j][i]와 다를 수 있다. 모든 도시간의 비용은 양의 정수이다. W[i][i]는 항상 0이다. 경우에 따라서 도시 i에서 도시 j로 갈 수 없는 경우도 있으며 이럴 경우 W[i][j]=0이라고 하자.

N과 비용 행렬이 주어졌을 때, 가장 적은 비용을 들이는 외판원의 순회 여행 경로를 구하는 프로그램을 작성하시오.

-풀이 (최수지)

```python
import sys
input=sys.stdin.readline
sys.setrecursionlimit(10000)
INF=sys.maxsize
n=int(input())
adj=[list(map(int,input().split()))for _ in range(n)]
dp=[[INF]*(1<<n)for _ in range(n)]
def dfs(current,visit):
    if visit==(1<<n)-1:
        if adj[current][0]==0:
            return INF
        else:
            return adj[current][0]
    if dp[current][visit]!=INF:
        return dp[current][visit]
    for i in range(1,n):
        if not visit&(1<<i) and adj[current][i]!=0:
            dp[current][visit]=min(dp[current][visit],dfs(i,visit|(1<<i))+adj[current][i])
    return dp[current][visit]
print(dfs(0,1))
```

6) 1520번 내리막 길

내리막 길 성공 출처 분류

☆

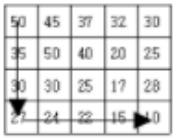
골드 IV

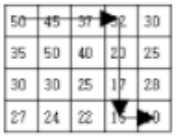
시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
2 초	128 MB	37438	9946	7141	28.156%

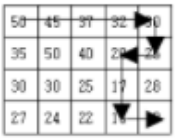
문제

여행을 떠난 세준이는 지도를 하나 구하였다. 이 지도는 아래 그림과 같이 직사각형 모양이며 여러 칸으로 나뉘어져 있다. 한 칸은 한 지점을 나타내는데 각 칸에는 그 지점의 높이가 쓰여 있으며, 각 지점 사이의 이동은 지도에서 상하좌우 이웃한 곳끼리만 가능하다.

현재 제일 왼쪽 위 칸이 나타내는 지점에 있는 세준이는 제일 오른쪽 아래 칸이 나타내는 지점으로 가려고 한다. 그런데 가능한 힘을 적게 들이고 싶어 항상 높이가 더 낮은 지점으로만 이동하여 목표 지점까지 가고자 한다. 위와 같은 지도에서는 다음과 같은 세 가지 경로가 가능하다.







지도가 주어질 때 이와 같이 제일 왼쪽 위 지점에서 출발하여 제일 오른쪽 아래 지점까지 항상 내리막길로만 이동하는 경로의 개수를 구하는 프로그램을 작성하시오.

-풀이 (김도연)

```python
import sys
input = sys.stdin.readline
sys.setrecursionlimit(10000)
dx = [1, -1, 0, 0]
dy = [0, 0, -1, 1]
def dfs(x, y):
    if x == 0 and y == 0:
        return 1
    if dp[x][y] == -1:
        dp[x][y] = 0
        for i in range(4):
            nx = x + dx[i]
            ny = y + dy[i]
            if 0 <= nx < m and 0 <= ny < n:
                if s[x][y] < s[nx][ny]:
                    dp[x][y] += dfs(nx, ny)
    return dp[x][y]
m, n = map(int, input().split())
s = [list(map(int, input().split())) for i in range(m)]
dp = [[-1] * n for i in range(m)]
print(dfs(m - 1, n - 1))
```

3. 정리
깃허브 인증

2021.03.15 ~

☐ 동적 프로그래밍(DP)

**공동문제 : 1520(내리막 길) **

이름	2839	2748	1912	11726	2098	1520
김도연	3/17	3/18	3/21	3/21	PASS	3/22
안희승	3/15	3/16	3/18	3/18	3/22	3/22
최수지	3/15	3/16	3/17	3/20	3/20	3/21
이윤희	3/15	3/16	3/19	3/19	PASS	3/21

* 학습활동 사진(필수)
03.29 오프라인 스터디 활동 인증
김도연 학생이 들고있는 핸드폰 화면에 활동시간 표기