

*DFS: Depth First Search(깊이 우선 탐색)

1) DFS는 스택 기반의 재귀함수를 이용하여 Back Tracking을 한다.

2) 재귀함수는 스택 구조이므로, x-1을 순서대로 1부터 출력하고 싶으면 x를 print하는 것과 DFS(x-1)의 위치를 바꾼다.

-DFS(3)을 수행하려고 Stack에 넣었을 때, DFS(2)가 Stack에 들어가고, DFS(1)이 Stack에 들어가서 DFS(n)이 비로소 끝나고 LIFO라서 DFS(1)부터 수행

3) 그 후 stack이 비었을 때 메인으로 와서 함수가 끝나게 됨

```
input.txt  AAp.py*  X
#깊이 우선 탐색
#재귀 함수이용

import sys
sys.stdin = open("input.txt", "rt")

def DFS(x):
    if x > 0: #무한루프 방지
        DFS(x-1)
        print(x, end = ' ') #옆으로 출력
        #스택구조이므로 순서대로 출력

if __name__ == "__main__": #main 함수
    n = int(input())
    DFS(n)
```

4) DFS는 상태트리를 구성하고 그것을 재귀함수로 잘 출력하면 할 수 있다.

1. 재귀함수란?(이진수 출력)

10진수 N이 입력되면 2진수로 변환하여 출력하는 프로그램을 작성하세요. 단 재귀함수를 이용해서 출력해야 합니다.

■ 입력설명

첫 번째 줄에 10진수 N($1 \leq N \leq 1,000$)이 주어집니다.

■ 출력설명

첫 번째 줄에 이진수를 출력하세요.

■ 입력예제 1

11

출력예제 1

1011

```
*untitled*
File Edit Format Run Options Window Help
#깊이 우선 탐색
#재귀 함수이용
#if-else로 종료지점 사용

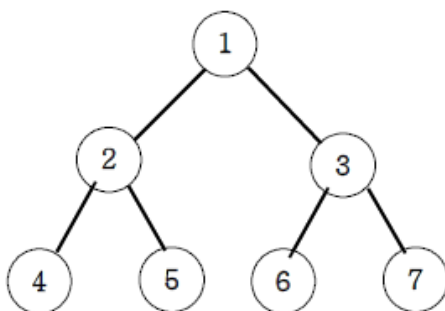
import sys
#sys.stdin = open("input.txt", "rt")

def DFS(x):
    if x == 0:
        return #함수 종료
    else:
        DFS(x // 2)
        print(x % 2, end= ' ')

if __name__ == "__main__": #main 함수
    n = int(input())
    DFS(n)
```

- 1) n을 입력받아 깊이우선탐색을 수행한다.
- 2) x가 0이면 함수를 종료하고, 그렇지 않다면 x를 2로 나눈 몫을 가지고 계속 DFS를 수행한다.
- 3) DFS가 수행하는 내용은 x를 2로 나눈 나머지를 반환하는 것이다.

2. 이진트리순회(깊이우선탐색)



전위순회 출력 : 1 2 4 5 3 6 7

중위순회 출력 : 4 2 5 1 6 3 7

후위순회 출력 : 4 5 2 6 7 3 1

1) 이진트리의 기본구조: 부모노드-왼쪽 자식노드-오른쪽 자식노드

-전위순회 출력: 부모(root)-왼쪽-오른쪽 => 깊이 우선 탐색에 주로 쓰이는 기법

=> 부모 노드가 1일 때, 왼쪽 자식 노드는 (부모노드*2), 오른쪽 자식 노드는 (부모노드*2+1)값을 갖는다.

```
def DFS(v):  
    if v>7:  
        return #종착지점  
    else:  
        print(v, end=' ')  
        DFS(v*2)#왼쪽 자식 노드  
        DFS(v*2+1)#오른쪽 자식 노드  
  
if __name__ == "__main__":  
    DFS(1)
```

-중위순회 출력: 왼쪽-부모(root)-오른쪽

```
def DFS(v):  
    if v>7:  
        return #종착지점  
    else:  
        DFS(v*2)#왼쪽 자식 노드  
        print(v, end=' ')  
        DFS(v*2+1)#오른쪽 자식 노드  
  
if __name__ == "__main__":  
    DFS(1)
```

-후위순회 출력: 왼쪽-오른쪽-부모(root)

```
def DFS(v):
    if v>7:
        return #종착지점
    else:
        DFS(v*2) #왼쪽 자식 노드
        DFS(v*2+1) #오른쪽 자식 노드
        print(v, end=' ')

if __name__ == "__main__":
    DFS(1)
```

3. 부분집합 구하기(DFS)

자연수 N 이 주어지면 1부터 N 까지의 원소를 갖는 집합의 부분집합을 모두 출력하는 프로그램을 작성하세요.

■ 입력설명

첫 번째 줄에 자연수 $N(1 \leq N \leq 10)$ 이 주어집니다.

■ 출력설명

첫 번째 줄부터 각 줄에 하나씩 아래와 같이 출력한다. 출력순서는 깊이우선탐색 전위순회방식으로 출력합니다. 단 공집합은 출력하지 않습니다.

■ 입력예제 1

3

■ 출력예제 1

1 2 3

1 2

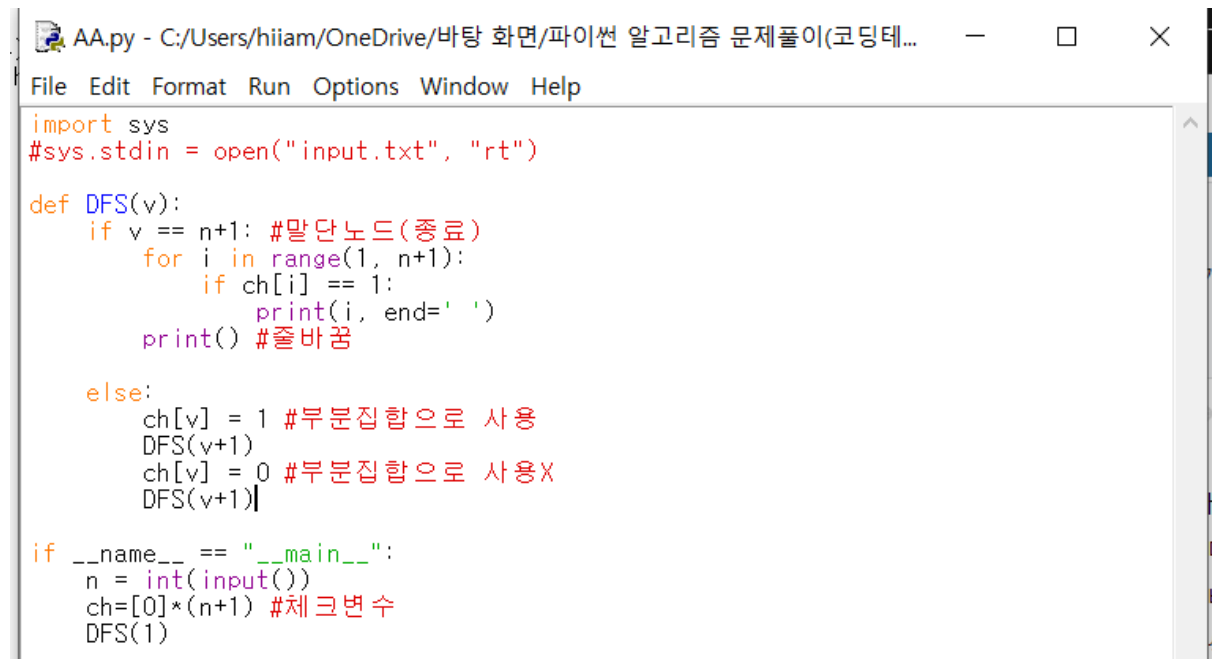
1 3

1

2 3

2

3



```
import sys
#sys.stdin = open("input.txt", "rt")

def DFS(v):
    if v == n+1: #말단노드(종료)
        for i in range(1, n+1):
            if ch[i] == 1:
                print(i, end=' ')
            print() #줄바꿈
    else:
        ch[v] = 1 #부분집합으로 사용
        DFS(v+1)
        ch[v] = 0 #부분집합으로 사용X
        DFS(v+1)

if __name__ == "__main__":
    n = int(input())
    ch=[0]*(n+1) #체크변수
    DFS(1)
```

4. 합이 같은 부분집합(DFS : 아마존 인터뷰)

N개의 원소로 구성된 자연수 집합이 주어지면, 이 집합을 두 개의 부분집합으로 나누었을 때 두 부분집합의 원소의 합이 서로 같은 경우가 존재하면 "YES"를 출력하고, 그렇지 않으면 "NO"를 출력하는 프로그램을 작성하세요. 예를 들어 {1, 3, 5, 6, 7, 10}이 입력되면 {1, 3, 5, 7} = {6, 10} 으로 두 부분집합의 합이 16으로 같은 경우가 존재하는 것을 알 수 있다.

■ 입력설명

첫 번째 줄에 자연수 N($1 \leq N \leq 10$)이 주어집니다.

두 번째 줄에 집합의 원소 N개가 주어진다. 각 원소는 중복되지 않는다.

■ 출력설명

첫 번째 줄에 "YES" 또는 "NO"를 출력한다.

■ 입력예제 1

6

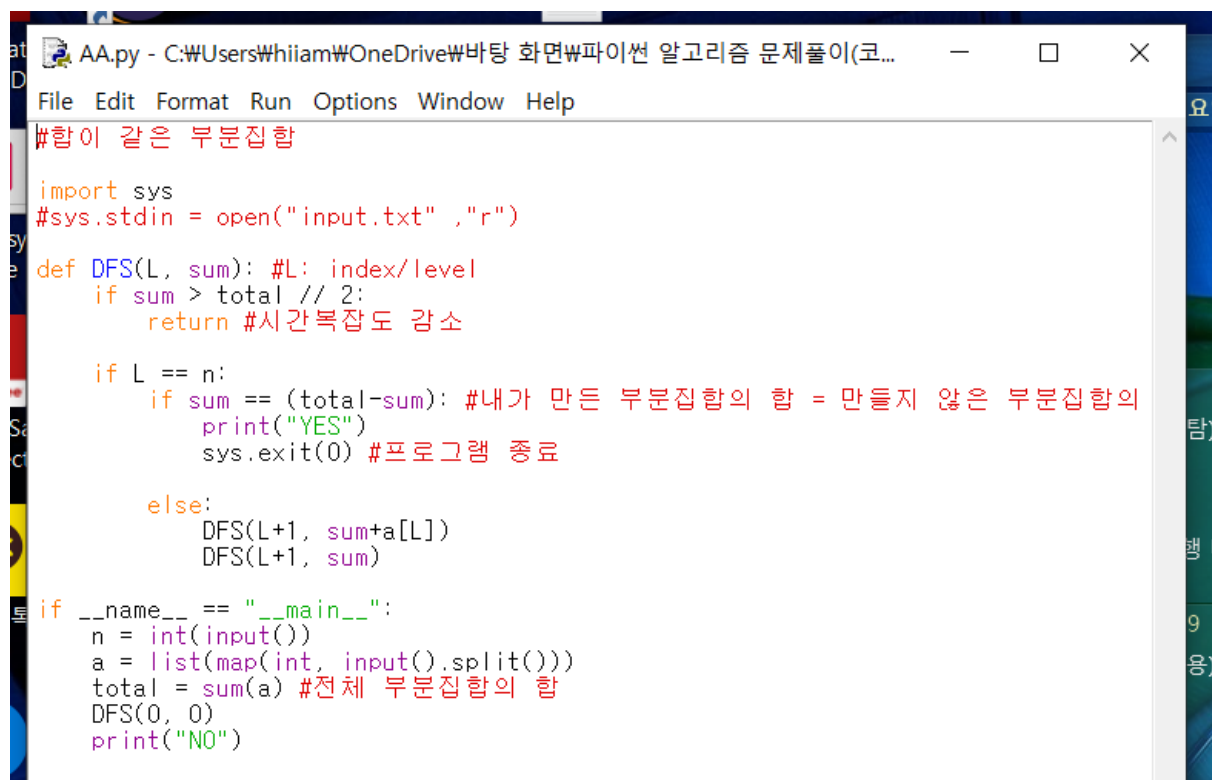
1 3 5 6 7 10

▣ 출력예제 1

YES

-전체 부분집합의 합에서 내가 만든 부분집합의 합을 빼주었을 때 또 다른 부분집합의 합이 되었다면 YES, 아니라면 NO를 출력

**틀리게 나옴



```
AA.py - C:\Users\Whiiam\OneDrive\바탕 화면\파이썬 알고리즘 문제풀이(코...
File Edit Format Run Options Window Help
#합이 같은 부분집합
import sys
#sys.stdin = open("input.txt", "r")
def DFS(L, sum): #L: index/level
    if sum > total // 2:
        return #시간복잡도 감소

    if L == n:
        if sum == (total-sum): #내가 만든 부분집합의 합 = 만들지 않은 부분집합의
            print("YES")
            sys.exit(0) #프로그램 종료
        else:
            DFS(L+1, sum+a[L])
            DFS(L+1, sum)

if __name__ == "__main__":
    n = int(input())
    a = list(map(int, input().split()))
    total = sum(a) #전체 부분집합의 합
    DFS(0, 0)
    print("NO")
```