

# 20일 (일)

## 1. 수 나열하기 - 등차+등비 수열 혼합 구현

for 문을 이용하여 직접 곱하고 더해준다.

## 2. 최소공배수 구현방법

```
a, b, c = map(int, input().split())  
  
d = 1  
  
while d%a != 0 or d%b != 0 or d%c != 0:  
    d += 1  
  
print(d)
```

d 는 a, b, c의 최소공배수 d에 각 수를 나누어 모두 0이 나올 때까지 d 를 증가해 나간다.

## 3. abs( ) 절대값 구하는 메소드

보통 거리 구할 때 많이 이용한다.

## 4. 딕셔너리 (사전) 의 키에 숫자를 넣을 때는 ' , "' 를 안 써도 된다.

## 5. 코드업 문제 기초 100제

#6087 등차수열에서 n번째 숫자 구하기

```
a, d, n = map(int, input().split())  
  
answer = a + (n-1)*d  
  
print(answer)
```

#6088 등비수열에서 n번째 숫자 구하기

```
a, r, n = map(int, input().split())  
  
answer = a*r**(n-1)  
  
print(answer)
```

#6089 등차수열 등비수열 조합된 수열 구하기 for문 이용

```
a, m, d, n = map(int, input().split())  
  
for i in range(n-1):  
    answer = a*m+d  
    a = answer  
  
print(a)
```

#6091 최소공배수 구현방법

```
a, b, c = map(int, input().split())  
  
d = 1  
  
while d%a != 0 or d%b != 0 or d%c != 0:  
    d += 1  
  
print(d)
```

```
#6092 이상한 출석번호 부르기 - 23번까지 번호에서 불린 횟수 출력하기
n = int(input())
```

```
call_number = list(map(int, input().split()))
```

```
array = [0]*23
```

```
for i in call_number:
    array[i-1] += 1
```

```
for i in array:
    print(i, end=' ')
```

```
#6093 이상한 출석번호 부르기 2 - 거꾸로 다시 부르기
```

```
n = int(input())
```

```
call_numbers = list(map(int, input().split()))
```

```
r_call_numbers = []
```

```
for i in range(1, n+1):
    r_call_numbers.append(call_numbers[-i])
```

```
for number in r_call_numbers:
    print(number, end=' ')
```

```
#이렇게도 가능 더 간결한 버전
```

```
for i in range(n-1, -1, -1) :
    print(a[i], end=' ')
```

```
#6094 이상한 출석번호 부르기 3 - 제일 빠른 번호 출력하기
```

```
#내 풀이
```

```
n = int(input())
```

```
call_num = list(map(int, input().split()))
```

```
answer = sorted(call_num)
```

```
print(answer[0])
```

```
#답안지 풀이
```

```
n = int(input())
```

```
a = input().split()
```

```
for i in range(n) :
    a[i] = int(a[i])
```

```

min = a[0]
for i in range(0, n) :
    if a[i] < min :
        min = a[i]

print(min)

#6095

n = int(input())

#바둑판 초기화
d = []

for i in range(20):
    d.append([])
    for j in range(20):
        d[i].append(0)

#바둑판 흰돌 놓기
for i in range(n):
    x, y = input().split()
    d[int(x)][int(y)] = 1

#바둑판 출력하기
for i in range(1, 20):
    for j in range(1, 20):
        print(d[i][j], end= ' ')
    print()

```

## 6. 프로그래머스 1단계 문제 풀이 - (2016년, 키패드 누르기)

### 2016년

## 2016년

### 문제 설명

2016년 1월 1일은 금요일입니다. 2016년 a월 b일은 무슨 요일일까요? 두 수 a ,b를 입력받아 2016년 a월 b일이 무슨 요일인지 리턴하는 함수, solution을 완성하세요. 요일의 이름은 일요일부터 토요일까지 각각 `SUN,MON,TUE,WED,THU,FRI,SAT`

입니다. 예를 들어 a=5, b=24라면 5월 24일은 화요일이므로 문자열 "TUE"를 반환하세요.

### 제한 조건

- 2016년은 윤년입니다.
- 2016년 a월 b일은 실제로 있는 날입니다. (13월 26일이나 2월 45일같은 날짜는 주어지지 않습니다)

### 입출력 예

a	b	result
5	24	"TUE"

### #나의 풀이

```
def solution(a, b):  
  
    # 누적일 수 설정  
    months = {  
        1 : 0, 2 : 31, 3 : 60, 4 : 91,  
        5 : 121, 6 : 152, 7 : 182, 8 : 213,  
        9 : 244, 10 : 274, 11 : 305, 12 : 335  
    }  
  
    weeks = ['FRI', 'SAT', 'SUN', 'MON', 'TUE', 'WED', 'THU']  
    days = ((months[a] + b)-1) % 7  
    answer = weeks[days]  
    return answer
```

#다른 사람 풀이 - 더 간단

```
def getDayName(a,b):  
    months = [31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]  
    days = ['FRI', 'SAT', 'SUN', 'MON', 'TUE', 'WED', 'THU']  
    return days[(sum(months[:a-1])+b-1)%7]
```

#아래 코드는 테스트를 위한 출력 코드입니다.

```
print(getDayName(5,24))
```

## 키패드 누르기

## [카카오 인턴] 키패드 누르기

### 문제 설명

스마트폰 전화 키패드의 각 칸에 다음과 같이 숫자들이 적혀 있습니다.



## [카카오 인턴] 키패드 누르기

이 전화 키패드에서 왼손과 오른손의 엄지손가락만을 이용해서 숫자만을 입력하려고 합니다.

맨 처음 왼손 엄지손가락은 \* 키패드에 오른손 엄지손가락은 # 키패드 위치에서 시작하며, 엄지손가락을 사용하는 규칙은 다음과 같습니다.

1. 엄지손가락은 상하좌우 4가지 방향으로만 이동할 수 있으며 키패드 이동 한 칸은 거리로 1에 해당합니다.
2. 왼쪽 열의 3개의 숫자 1, 4, 7 을 입력할 때는 왼손 엄지손가락을 사용합니다.
3. 오른쪽 열의 3개의 숫자 3, 6, 9 를 입력할 때는 오른손 엄지손가락을 사용합니다.
4. 가운데 열의 4개의 숫자 2, 5, 8, 0 을 입력할 때는 두 엄지손가락의 현재 키패드의 위치에서 더 가까운 엄지손가락을 사용합니다.  
4-1. 만약 두 엄지손가락의 거리가 같다면, 오른손잡이는 오른손 엄지손가락, 왼손잡이는 왼손 엄지손가락을 사용합니다.

순서대로 누를 번호가 담긴 배열 numbers, 왼손잡이인지 오른손잡이인 지를 나타내는 문자열 hand 가 매개변수로 주어질 때, 각 번호를 누른 엄지손가락이 왼손인 지 오른손인 지를 나타내는 연속된 문자열 형태로 return 하도록 solution 함수를 완성해주세요.

### [제한사항]

- numbers 배열의 크기는 1 이상 1,000 이하입니다.
- numbers 배열 원소의 값은 0 이상 9 이하인 정수입니다.
- hand는 "left" 또는 "right" 입니다.
  - "left" 는 왼손잡이, "right" 는 오른손잡이를 의미합니다.
- 왼손 엄지손가락을 사용한 경우는 L, 오른손 엄지손가락을 사용한 경우는 R 을 순서대로 이어붙여 문자열 형태로 return 해주세요.

### 입출력 예

numbers	hand	result
[1, 3, 4, 5, 8, 2, 1, 4, 5, 9, 5]	"right"	"LRLLLRLLRRL"
[7, 0, 8, 2, 8, 3, 1, 5, 7, 6, 2]	"left"	"LRLRRLLLRRL"

```
def solution(numbers, hand):  
    answer = ''
```



```

#손가락 현재 위치 설정
tmp_r = '#'
tmp_l = '*'

#배열 내 숫자 탐색
for number in numbers:

    if number in [1, 4, 7]:
        answer += 'L'
        tmp_l = number

    elif number in [3, 6, 9]:
        answer += 'R'
        tmp_r = number

    elif number in [2, 5, 8, 0]:
        #손가락의 현재위치에서 숫자로부터 거리를 구해서 비교
        d_r = distance(tmp_r, number)
        d_l = distance(tmp_l, number)

        if d_r > d_l:
            answer += 'L'
            tmp_l = number

        elif d_r < d_l:
            answer += 'R'
            tmp_r = number

        elif d_r == d_l:

            if hand == 'right':
                answer += 'R'
                tmp_r = number

            else:
                answer += 'L'
                tmp_l = number

    return answer

def distance(fin_position, number):

    keypad = {'1': (0, 0), '2': (0, 1), '3': (0, 2),
              '4': (1, 0), '5': (1, 1), '6': (1, 2),
              '7': (2, 0), '8': (2, 1), '9': (2, 2),
              '*': (3, 0), '0': (3, 1), '#': (3, 2)}

    hand = str(hand)
    number = str(number)
    x_h, y_h = keypad[fin_position]
    x_n, y_n = keypad[number]

    return abs(x_h - x_n) + abs(y_h - y_n)

```

## 7. 유튜브 강의 - BOJ 로 처음 알고리즘 시작해서 공부했던 방법 공유 by 라매개발자

- 1) 백준 알고리즘 단계별로 풀기 정렬 12번까지 풀기
- 2) 강의 탭 들어가서 백준 알고리즘 기초1, 2 강의 안에 있는 문제 내역 보고 순서대로 풀기
- 3) 이 때쯤 웬만한것을 구현할 수 있는 자신감이 생기게 됨
- 4) 이제 중급으로 넘어가도 된다.

\*\* 문제를 고민해보다가 30분 이상 지나면 바로 코드를 찾아 보고 이해한다.

패턴을 외우고 문제가 보이면 패턴을 적고 어떻게 해결할 지 보면 풀린다.

외우고 문제를 많이 풀면 어쨌든 늘게 된다.