

# 코딩, Python으로 같이 시작해요



2주차 : 반복문, 함수

# 과제 설명-1

## <과제1 : if문 활용>

- if문 중첩을 활용하는 방법

```
a = int(input("Num1 : "))
b = int(input("Num2 : "))
c = int(input("Num3:  "))
if a>=b :
    if b>=c: #a>b>c
        print(a,">",b,">",c)
    else:
        if a>=c : #a>c>b:
            print(a,">",c,">",b)
        else : #c>a>b
            print(c,">",a,">",b)
else:
    if a>=c : #b>a>c
        print(b,">",a,">",c)
    else:
        if b>=c: #b>c>a
            print(b,">",a,">",c)
        else : #c>b>a
            print(c,">",b,">",a)
```

- if문의 중첩 없이 변수 두 개를 바꾸는 방법을 이용.

```
if b>a:
    a,b = b,a
if c>a:
    a,c = c,a
if c>b:
    c,b = b,c
print(a,b,c)
```

## 과제 설명-2

<과제2 : 문자열 슬라이싱>

```
ID = input("주민등록번호를 입력하세요 : ")
a = ID[7]
print("\n생년월일 :", ID[:6])

if a == '1':
    print("성별 : 남")
elif a == '2':
    print("성별 : 여")
else:
    print("어느 시대 사람인가요?")
```

# 과제 설명-3

<과제3 : 리스트 슬라이싱>

```
movie_rank = ["어벤져스", "쥬라기 공원", "기생충", "옥자", "살인의 추억"]  
a,b = input("알고 싶은 영화 순위의 범위를 알려주세요 : ").split()  
a = int(a)  
b = int(b)  
print(movie_rank[a:b+1])
```

# 과제 설명-4

<과제4 : 프로그램 흐름의 분기! 메뉴 선정 프로그램>

이 과제는 사람마다 모두 코드가 다를 수 있어요! 그냥 멘토는 저렇게 했구나 속 참고만 하시면 됩니다.

```
money = int(input("돈 : "))

if money >= 4000:
    sweet = input("달달한 게 땡기시나요?(Y/N) : ")
    if sweet == "Y":
        cream = input("생크림 좋아하시나요?(Y/N) : ")
        if cream == "Y":
            print("\n오레오 프라푸치노")
        else:
            print("\n알로에 스무디")
    else:
        print("\n카페 라떼")
else:
    sweet = input("달달한 게 땡기시나요?(Y/N) : ")
    if sweet == "Y":
        print("\n바닐라 라떼")
    else:
        hot = input("뜨거운 음료가 땡기나요?(Y/N) : ")
        if hot == "Y":
            print("\n녹차")
        else:
            print("\n아이스 아메리카노")
```

# 과제 설명-5

<과제5 : 경우를 나눠서 생각하는 문제. 시간 계산 프로그램>

```
hour, minute = map(int, input("내일 몇시에 일어나시나요? ").split(':'))

if minute >= 50:
    minute -= 50
elif hour != 0 and minute < 50:
    hour -= 1
    sub = 50 - minute
    minute = 60 - sub #minute+=10 과 동일
elif hour == 0 and minute < 50:
    hour = 23
    minute += 10

print(hour, "시", minute, "분에 알람을 맞춰 드릴게요!\n")
```

# 과제 설명-6

<과제6 : 경우를 나눠서 조건에 해당하는 코드를 실행하는 프로그램>

```
ID = "ektmf7890"
password = "&password&"
```

```
a = input("ID : ")
b = input("password : ")
```

```
if a == ID and b == password:
    print(ID+"님, 환영합니다!")
elif a == ID and b != password:
    print("비밀번호를 까먹으셨나요?\n")
    ans = input("힌트를 드릴까요?(y/n) : ")
    if ans == 'y':
        print(password[:4]+"****")
    else:
        print("로그인에 실패했습니다.")
else :
    print("존재하지 않는 계정입니다.\n로그인에 실패하였습니다.")
```

# 과제 설명-6 심화

<과제6 심화 : 원하는 출력을 얻기 위해 코드의 구성을 수정하는 연습>

```
ID = "ektmf7890"
password = "&password&"

a = input("ID : ")

if a==ID:
    b= input("password : ")
    if b==password:
        print(ID+"님, 환영합니다!")
    else:
        print("비밀번호를 까먹으셨나요?\n")
        ans = input("힌트를 드릴까요?(y/n) : ")
        if ans == 'y':
            print(password[:4]+"****")
        else:
            print("로그인에 실패했습니다.")
else:
    print("존재하지 않는 계정입니다.\n로그인에 실패했습니다.")
```



# 반복문 - for

for 변수 in ■■■ :

<네모에 들어갈 수 있는 값>

- range()
- list
- string

1. range()와 사용하는 방법

```
for i in range(31):  
    if i%3 == 0:  
        print(i)
```

0 3 6 9 12 15 18 21 24 27 30

2. list와 사용하는 방법

```
my_list = ['a', 'b', 'c', 'd', 'e', 'f']  
for i in my_list:  
    print(i, end = ' ')
```

a b c d e f

# 반복문 - for

## 3. 문자열과 사용하는 방법

```
my_string = "forif=for+if"
for i in my_string:
    print(i, end=' ')
```

```
f o r i f = f o r + i f
```

```
a, b = input().split()
a = int(a)
b = int(b)

total = 0
for i in range(a, b+1):
    total+=i
print(total)
```

```
2 5
14
```

# for문 실습

range() 함수를 이용해 원하는 범위 내의 숫자를 반복하고, 이를 이용해 구구단을 출력합니다.

```
num = int(input("정수 입력 : "))
for i in range(1,10):
    print(f"{num} x {i} = {num*i}")
```

```
정수 입력 : 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
```

학생들의 성적을 담은 리스트를 이용해 전체 평균 성적을 계산합니다.

```
scores = [90, 45, 72, 84, 65, 57]

total_sum = 0
for score in scores:
    total_sum += score
print("전체 평균 :", total_sum//len(scores))
```

```
전체 평균 : 68
```

# 반복문 - while

*while 조건식:*

*실행할 코드*

#코드 어딘가에 조건을 변화시키는 모드가 존재해야 무한 루프를 방지!

```
i=0
while i < 5:
    print("Happy!", i)
    i+=1
```

```
Happy! 0
Happy! 1
Happy! 2
Happy! 3
Happy! 4
```

```
i=1
while i<=5:
    print("Happy!",i)
    i+=1
```

```
Happy! 1
Happy! 2
Happy! 3
Happy! 4
Happy! 5
```

```
n = int(input("반복 횟수 : "))
while n>0:
    print("Happy!", n)
    n-=1
```

```
반복 횟수 : 3
Happy! 3
Happy! 2
Happy! 1
```

# 반복문 - while

<반복 횟수가 지정되지 않았을 때 while문을 사용하는 방법!>

```
password = "이육만세"  
attempt = str()  
while attempt != password:  
    attempt = input()  
print("비밀번호 일치")
```

이육  
이육만세?  
이육만세  
비밀번호 일치

```
import random  
  
a = 0  
while a != 2:  
    a = random.randint(1,6)  
    print(a)  
print("'2가 생성되어 반복문이 종료되었습니다.'")
```

5  
3  
1  
4  
6  
3  
2  
'2가 생성되어 반복문이 종료되었습니다.'

# break, continue

break

- 반복문을 즉시 빠져나온다.

```
while True:
    choice = int(input("1. Water\n2.Cola\n3.Beer\nYour choice(0 to end) : "))
    if choice == 1:
        print("you chose water")
    elif choice == 2:
        print("you chose cola")
    elif choice == 3:
        print("you chose beer")
    else:
        break
```

```
1. Water
2.Cola
3.Beer
Your choice(0 to end) : 1
you chose water
1. Water
2.Cola
3.Beer
Your choice(0 to end) : 0
```

# break, continue

## continue

- 반복문을 빠져나가지는 않는다.
- 뒤에 이어지는 코드를 실행하지 않고 다음 루프를 실행하러, for 또는 while이 있는 부분으로 이동한다.

```
for i in range(1,20):  
    if i%2 == 0:  
        continue  
    print(i, end = ' ')
```

1 3 5 7 9 11 13 15 17 19

# 반복문 실습1

<사칙연산 계산기> calculator.py 라고 저장해주세요!  
잠시 후 실습에서 다시 사용합니다!!!!

```
while True:
    print("Menu".center(25, '-'))
    print("1.Add 2.Sub 3.Mul 4.Div\n")
    choice = int(input("Your Choice(0 to end) : "))
    if choice == 0:
        break
    a, b = input("Enter two numbers : ").split()
    a = int(a)
    b = int(b)
    if choice == 1:
        print("Result :", a+b)
    if choice == 2:
        print("Result : ", a-b)
    if choice == 3:
        print("Result : ", a*b)
    if choice == 4:
        print("Result : ", a/b)
```



# 반복문 실습1

<사칙연산 계산기> calculator.py 라고 저장해주세요!  
잠시 후 실습에서 다시 사용합니다!!!!

```
while True:
    print("Menu".center(25, '-'))
    print("1.Add 2.Sub 3.Mul 4.Div\n")
    choice = int(input("Your Choice(0 to end) : "))
    if choice == 0:
        break
    a, b = input("Enter two numbers : ").split()
    a = int(a)
    b = int(b)
    if choice == 1:
        print("Result :", a+b)
    if choice == 2:
        print("Result : ", a-b)
    if choice == 3:
        print("Result : ", a*b)
    if choice == 4:
        print("Result : ", a/b)
```

```
-----Menu-----
1.Add 2.Sub 3.Mul 4.Div

Your Choice(0 to end) : 1
Enter two numbers : 2 3
Result : 5

-----Menu-----
1.Add 2.Sub 3.Mul 4.Div

Your Choice(0 to end) : 3
Enter two numbers : 3 4
Result : 12

-----Menu-----
1.Add 2.Sub 3.Mul 4.Div

Your Choice(0 to end) : 0
>>> |
```

# 반복문 실습2

## <별 찍기>

: 별 찍기 문제는 코딩에서 반복문의 중첩을 연습하는 예제로 주로 활용되지만, 사실 파이썬에서는 반복문을 중첩하지 않는 방법도 존재합니다. 두 가지 방법 모두 알려 드릴게요! 별 찍기 과제에서는 반복문 중첩 사용해주세요!  
그걸 연습하기 위한 문제이기 때문이죠 ㅎㅎ

```
for i in range(5):  
    for j in range(i+1):  
        print("*", end=' ')  
    print()
```

```
|for i in range(5):  
    print("*" * (i+1))
```

```
*  
**  
***  
****  
*****
```

# 반복문 실습 3

<최대공약수를 구하는 유클리드 알고리즘 – 반복문 구현>

- 유클리드 호제법

: 두 수의 최대공약수를 구하는 알고리즘

1. 임의의 두 자연수  $a, b$ 가 주어질 때, 큰 값은  $a$ , 작은 값은  $b$ !
2.  $a$ 를  $b$ 로 나눈 나머지를  $n$ 에 저장한다.
3.  $n==0$  ->  $b$ 가 최대공약수
4.  $n!=0$  ->  $a$ 에  $b$ 의 값을 넣고  $n$ 을  $b$ 에 대입한 후 위 과정을 다시 반복한다.

```
a, b = map(int, input("a,b = ").split())
```

```
if b>a:
    b,a = a,b
n=1
while n!=0:
    n = a%b
    a=b
    b=n
```

```
print(f"GCD : {a}")
```

```
a,b = 12 15
GCD : 3
```

# 함수

<함수: fuction>

- 특정 기능을 수행하는 코드의 모음
- 코드의 다른 부분에서 필요할 때마다 함수를 불러다 사용할 수 있음

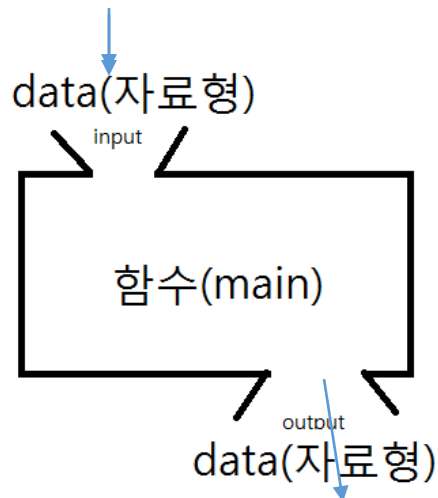
: 함수를 만드는 것 -> 함수의 정의

: 함수를 불러다 사용하는 것 -> 함수 호출

#주의사항#

함수를 만들기 전에 함수를 호출해서 사용하려고 하면 에러가 발생합니다!

<파이썬에서 함수의 구조>



```
def 함수_이름 (매개변수1, 매개변수2, .... 매개변수 n):  
    실행할 코드..  
    return 반환 값
```

- 매개변수 = 함수의 input!
- 반환 값 = 함수의 output!

# 함수

<함수의 다양한 유형들>

1. 매개변수 X, 반환 값 X

```
def list_maker():  
    a = []  
    print("I made a list", a)
```

I made a list []

list\_maker()

2. 매개변수 X, 반환 값 0

```
def list_maker():  
    a = []  
    return a
```

[]  
>>> |

```
my_list = list_maker()  
print(my_list)
```

# 함수

<함수의 다양한 유형들>

3. 매개변수 0, 반환 값 X

```
def list_maker():  
    a = []  
    return a
```

```
def list_add_name(name, my_list):  
    my_list.append(name)
```

```
my_list = list_maker()  
list_add_name("Dayoung", my_list)  
print(my_list)
```

['Dayoung']

4. 매개변수 0, 반환 값 0

```
def list_maker():  
    a = []  
    return a
```

```
def list_add_name(name, my_list):  
    my_list.append(name)
```

```
def list_get_name(my_list):  
    return my_list[0]
```

```
my_list = list_maker()  
  
list_add_name("Dayoung", my_list)  
  
a = list_get_name(my_list)  
  
print(a)
```

| Dayoung

# 함수

<return에 관하여>

# return은 값을 반환할 때 뿐만 아니라, 함수를 중간에서 빠져 나오기 위한 용도로 많이 사용됩니다.

# return으로 값을 반환하거나 함수를 빠져 나오게 되면, 처음에 함수를 호출했던 위치로 프로그램이 되돌아갑니다!

```
def scholarship(GPA):  
    if GPA < 2.5:  
        return  
    print("장학금 수혜 대상입니다.")
```

```
>>> scholarship(2.3)  
>>> scholarship(3.5)  
장학금 수혜 대상입니다.
```

# 함수

<함수 안에서 선언한 변수의 효력 범위>

# 함수 안에서 새로 만든 매개변수는 함수 안에서만 사용되는 함수만의 변수이다!

# 함수 안의 변수와 함수 밖의 변수는 이름이 같아도 서로 전혀 관련이 없는 변수임을 기억하자!

```
def change_age(age):  
    age+=1
```

```
age = 22  
change_age(age)  
print(age)
```

```
22
```

```
>>> |
```



# 함수 실습

<사칙연산 – 함수로 구현>    **아까 저장한 calculator.py를 열어서 아래와 같이 수정하세요!**



```
while True:
    print("Menu.".center(25, '-'))
    print("1.Add 2.Sub 3.Mul 4.Div\n")
    choice = int(input("Your choice(0 to end) : "))
    if choice == 0:
        break
    a,b = input("Enter two numbers : ").split()
    a = int(a)
    b = int(b)
    if choice == 1:
        print("Result :", add(a,b))
    if choice == 2:
        print("Result :", sub(a,b))
    if choice == 3:
        print("Result :", mul(a,b))
    if choice == 4:
        print("Result :", div(a,b))
```

← 이렇게 함수를 호출했을 때 계산 값이 반환되도록  
add, sub, mul, div 4개의 함수를 정의하세요.  
핑크색 박스 위치에 함수를 만드세요!

# 함수 실습

<사칙연산 – 함수로 구현>    **아까 저장한 calculator.py를 열어서 아래와 같이 수정하세요!**

```
def add (a, b):  
    return a+b  
def sub (a, b):  
    return a-b  
def mul (a, b):  
    return a*b  
def div (a, b):  
    return a/b  
  
while True:  
    print("Menu.".center(25, '-'))  
    print("1.Add 2.Sub 3.Mul 4.Div\n")  
    choice = int(input("Your choice(0 to end) : "))  
    if choice == 0:  
        break  
    a,b = input("Enter two numbers : ").split()  
    a = int(a)  
    b = int(b)  
    if choice == 1:  
        print("Result :", add(a,b))  
    if choice == 2:  
        print("Result :", sub(a,b))  
    if choice == 3:  
        print("Result :", mul(a,b))  
    if choice == 4:  
        print("Result :", div(a,b))
```

← 이렇게 함수를 호출했을 때 계산 값이 반환되도록  
add, sub, mul, div 4개의 함수를 정의하세요.  
핑크색 박스 위치에 함수를 만드세요!

# 재귀함수

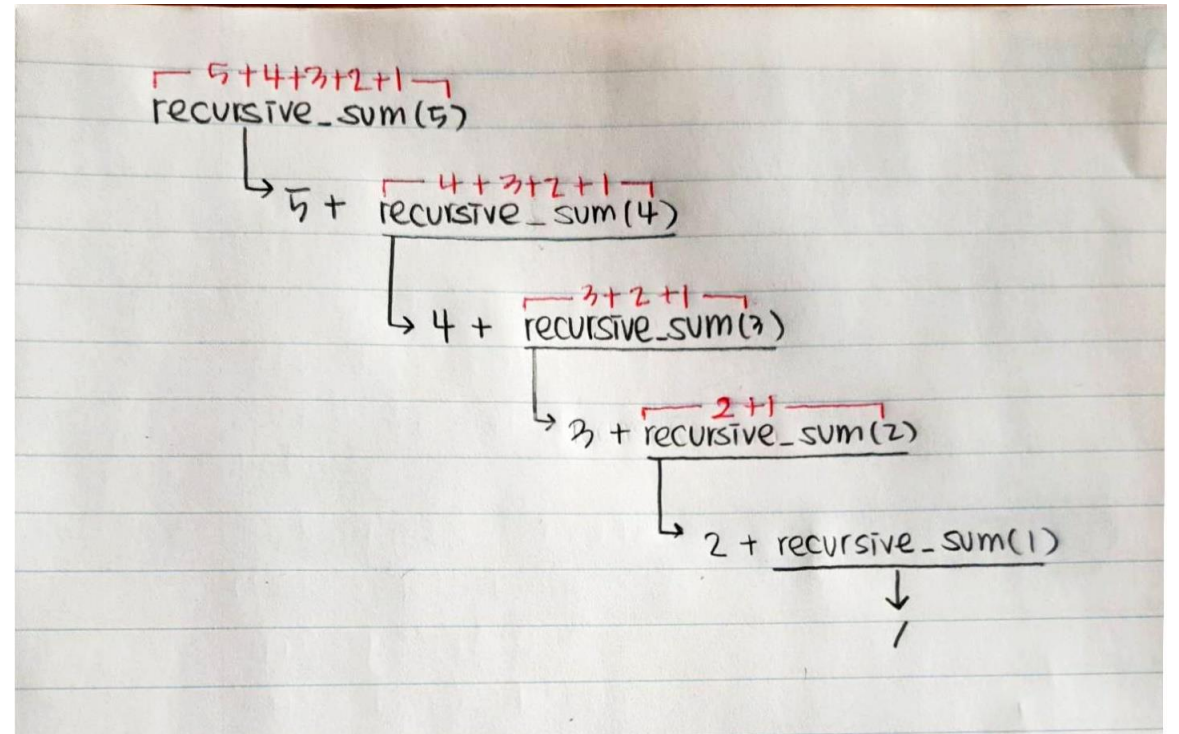
<재귀함수>

: 함수 안에서 함수 자기 자신을 호출하는 방식

```
def recursive_sum(n):  
    if n == 1:  
        return 1  
    return n + recursive_sum(n-1)
```

```
n = int(input())  
print(recursive_sum(n))
```

```
5  
15  
>>> |
```




# 재귀함수 실습

<팩토리얼>

:  $n! = n * (n-1) * (n-2) * \dots * 1$

:  $4! = 24, 5! = 120 \dots$

```
def fact(n):  
    if n == 1:  
        return 1  
    return 
```

```
n = int(input())  
print(fact(n))|
```

```
4  
24  
>>> |
```

앞의 팩토리얼 예제와 똑같은 형태의 코드입니다!

잘 이해가 안되시면 앞 슬라이드에 제가 그려놓은 것처럼 그림을 그리면서 코드를 쫓 따라가 보세요.

저도 처음 코딩을 할 때 재귀함수가 너무 어려워서 계속 그림을 그리면서 공부했었습니다!

# 재귀함수 실습

<팩토리얼>

:  $n! = n * (n-1) * (n-2) * \dots * 1$

:  $4! = 24, 5! = 120 \dots$

```
def fact(n):  
    if n == 1:  
        return 1  
    return n*fact(n-1)
```

```
n = int(input())  
print(fact(n))
```

```
4  
24  
>>> |
```

앞의 팩토리얼 예제와 똑같은 형태의 코드입니다!

잘 이해가 안되시면 앞 슬라이드에 제가 그려놓은 것처럼 그림을 그리면서 코드를 쫓 따라가 보세요.

저도 처음 코딩을 할 때 재귀함수가 너무 어려워서 계속 그림을 그리면서 공부했었습니다!

# 과제 1

<공배수 처리하기>

: week2-1.py

: 1-99 까지의 자연수 중 3의 배수를 만나면 'A'를, 5의 배수를 만나면 'B'를 출력합니다.

: 3의 배수이면서 5의 배수인 수는 'AB'를 출력합니다.

: 예시) 15-> AB 출력

1

2

A

4

B

A

7

8

A

B

11

A

13

14

AB

## 과제 2

<최대공약수를 구하는 유클리드 알고리즘 – 재귀함수 구현>

: week2-2.py

:반복문 실습3에서 배운 유클리드 호제법을 재귀함수로 구현한다.

- 유클리드 호제법

: 두 수의 최대공약수를 구하는 알고리즘

1. 임의의 두 자연수  $a, b$ 가 주어질 때, 큰 값은  $a$ , 작은 값은  $b$ !
2.  $a$ 를  $b$ 로 나눈 나머지를  $n$ 에 저장한다.
3.  $n == 0 \rightarrow b$ 가 최대공약수
4.  $n \neq 0 \rightarrow a$ 에  $b$ 의 값을 넣고  $n$ 을  $b$ 에 대입한 후 위 과정을 다시 반복한다

# 과제 3

<별 찍기 약간 어려운 문제>

: week2-3.py

: 첫째 줄 부터 n번째 줄까지 차례대로 별을 출력한다.

: for문의 중첩을 사용해주세요! 그걸 공부하기 위한 문제입니다.

: 형식대로 별을 출력해주는 stars함수를 따로 정의하고, 함수를 호출하세요.

```
def stars(n):
```



```
n = int(input("정수 입력 : "))  
stars(n)|
```

정수 입력 : 3

```
*  
  
***  
  
*****
```

정수 입력 : 5

```
*  
  
***  
  
*****  
  
*****  
  
*****
```

정수 입력 : 4

```
*  
  
***  
  
*****  
  
*****
```



## 과제 4

<자판기> week2-4.py

- 사용자에게 돈을 입력 받는다.
- 메뉴를 예시와 같이 출력한다.
- 사용자가 1~4사이의 정수를 입력 시  
: 잔액이 가격보다 큰지 확인하고 아니면 문구를 출력한다.  
: 잔액이 가격보다 크거나 같으면 잔액에서 해당하는 액수를 차감하고 문구를 출력한다.
- 0,1,2,3,4 이외의 값을 입력하면 오류 문구를 띄우고 다시 입력을 받는다.
- while문을 사용해 반복해서 입력을 받다가 0을 입력하면 프로그램을 종료한다.  
단, 잔액이 2000원 미만일 때도 “You can’t buy anything with {잔액}”을 출력하고 프로그램을 종료한다.

## 과제 4

<출력 예시> : 참고해서 비슷하게만 만들어 보세요! 완벽하게 같지 않아도 됩니다.

```
Insert money : 10000
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : 1
```

```
You got a Cola!
Money : 8000
```

```
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : 2
```

```
You got a Sprite!
Money : 5500
```

```
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : 3
```

```
You got a Coffee!
Money : 2500
```

```
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : 4
```

```
Not enough money
Money : 2500
```

```
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : -1
```

```
Error, your choice should be between 0~4
Money : 2500
```

```
-----[MENU]-----
1. Cola          2000
2. Sprite        2500
3. Coffee        3000
4. MintChoco     4500
-----
Choice(0 to end) : 1
```

```
You got a Cola!
Money : 500
```

```
You can't buy anything with 500원
>>>
```

# 과제 5

<피보나치 수열>

week2-5.py

f(0) = 0      f(7) = 13  
f(1) = 1      f(8) = 21  
f(2) = 1      f(9) = 34  
f(3) = 2      f(10) = 55  
f(4) = 3  
f(5) = 5  
f(6) = 8

-> 입력한 점수에 해당하는 피보나치 수열을 출력합니다.

-> 주어진 코드는 이를 for문으로 구현했습니다. 여러분은 다음의 코드를  
참고해서 재귀함수로 피보나치 수열을 구하는 코드를 작성하세요.

```
def fibo(n):  
    a=0  
    b=1  
    for i in range(n):  
        temp = a  
        a = b  
        b = temp + b  
    return a
```

```
n = int(input())  
print(fibo(n))
```

```
7  
13  
>>> |
```

# 심화 과제

<369게임> week2adv.py

:백준 알고리즘 사이트 17614문제 <https://www.acmicpc.net/problem/17614>

## 문제

민수는 같은 반 친구들과 369게임을 하고 있다. 369게임은 여러 명이 원형으로 둘러 앉아 시작 위치의 사람이 1을 외치며 시작된다. 이후 시계방향으로 돌아가며 2, 3, 4와 같이 1씩 증가된 수가 자기 수가 된다. 순서대로 돌아오는 자기 수에 3, 6, 혹은 9가 포함되어 있지 않다면 그 수를 말해야 하며, 3, 6, 혹은 9가 포함되어 있으면 그 개수만큼 박수를 쳐야 한다. 이 규칙을 지키지 못하면 게임이 종료된다.

민수는 369게임이 N까지 규칙을 지키며 진행된다면 그때까지의 들은 박수의 횟수가 총 몇 번인지 궁금했다. 예를 들어  $N = 14$ 라면, 3, 6, 9, 13에서 각각 한 번의 박수를 치게 되므로 총 4회의 박수를 듣게 될 것이다.  $N = 36$ 이라면 3, 6, 9, 13, 16, 19, 23, 26, 29, 30, 31, 32, 33, 34, 35, 36에서 박수를 치게 되는데 33, 36에서는 각각 두 번 박수를 쳐야 하므로 총 18회가 된다. 1 이상의 정수 N에 대하여 369게임을 N까지 규칙을 지키며 진행된다면 그때까지 듣게 되는 박수의 총 횟수를 계산하여 출력하는 프로그램을 작성하시오.

## 입력

첫 번째 줄에 정수 N이 주어진다 ( $1 \leq N \leq 10^6$ ).

## 출력

박수의 총 횟수를 정수로 출력한다.

14

4

>>> |

36

18

>>> |

## 스터디 진행 일정

- 중간고사 일정에 관한 조사가 완료되는 대로 스터디 휴강 일정을 확정하고 공지하겠습니다.
- 3주차 수업의 내용은 과제 및 스터디의 난이도에 관한 여러분의 피드백(구글폼)을 참고해 결정하겠습니다.

# 과제 제출 안내

- 과제 제출은 **다음 주 금요일 자정** 까지 입니다!

(과제가 많고 어려울 수 있어서 넉넉하게 시간 드립니다.)

- 질문 많이 해주세요!

여러분이 질문을 많이 해주셔서 요즈음 정말 뿌듯합니다 ㅎㅎ

제출 이메일 : [ektmf7890@hanyang.ac.kr](mailto:ektmf7890@hanyang.ac.kr)