소프트웨어 입문 설계

실습 4주차-2

과제 4-2

김윤호 교수님

2021 봄학기

목차

- 과제 4-1 리뷰
- 과제 4-2 안내

과제 4-1 리뷰

- 1번 문항
  - 정수 값 하나를 입력 받음 (1 < n)
  - 1부터 n까지의 정수 중에서 2의 배수 혹은 3의 배수인 수를 모두 출력

(실행 예)

```
10⏎
2
3
4
6
8
9
10
```

```
n = int(input())


# 첫번째 방법
for i in range(1,n+1):
    if i % 2 == 0:                          # 2의 배수일 경우
        print(i)
    elif i % 3 == 0:                        # 3의 배수일 경우
        print(i)


# 두번째 방법
for i in range(1,n+1):
    if i % 2 == 0 or i % 3 == 0:            # 2의 배수나 3의 배수일 경우
        print(i)
```

과제 4-1 리뷰

- 2번 문항
  - 정수를 두 개 입력 받음
  - 두 수의 합, 차, 곱 등의 값을 출력
    - Add ,Sub, Mul, Div, Mod, printMsg 함수를 반드시 사용하여 작성

(실행 예)

```
10 ⏎
6 ⏎
sum: 16
difference: 4
product: 60
division: 1.6666666666666667
remainder: 4
completed
```

```
n = int(input())
n2 = int(input())

def Add(num1, num2):
    result = num1 + num2
    return result

def Sub(num1, num2):
    result = num1 - num2
    return result

def Mul(num1, num2):
    result = num1 * num2
    return result

def Div(num1, num2):
    result = num1 / num2
    return result

def Mod(num1, num2):
    result = num1 % num2
    return result

def printMsg():
    print("completed")

print("sum:" , Add(n,n2))
print("difference:" , Sub(n,n2))
print("product:" , Mul(n,n2))
print("division:" , Div(n,n2))
print("remainder:" , Mod(n,n2))
printMsg()
```

과제 4-1 리뷰

- 3번 문항
  - 정수를 하나를 입력 받음 (1 ≤ n ≤ 10)
  - 아래 두 함수에 정수 n 을 인자로 넘겨 addTotal()의 반환 값과 gMul의 값 출력
    - addTotal과 mulTotal 함수를 작성해야 하며, gMul이라는 global variable 이용

```
n = int(input())

def addTotal(num):
    sum = 0
    for i in range(1,num+1):
        sum += i
    return sum

def mulTotal(num):
    global gMul
    gMul = 1
    for i in range(1,num+1):
        gMul *= i

print("addTotal():",addTotal(n))
mulTotal(n)                              # gMul에 대한 곱셈이 이루어져야 하므로 함수 호출
print("gMul:",gMul)
```

(실행 예)

```
5⏎
addTotal(): 15
gMul: 120
```

과제 4-2

- 제출 안내
  - **제출 기한: 3월 31일 (수) 23시 59분** (git push 완료 시점 기준)
  - 프로젝트 하위에 *<과제번호>/<문제번호>/<답안 파일>* 구조를 갖도록 제출

- 프로젝트 구조 예시

```
+ 2021_ITE1014_2021000001
  + 4-2/
    + 1/
      - p1.py
    + 2/
      - p2.py
    + 3/
      - p3.py
```

과제 4-2

- 1번 문항
  - 정수 값을 하나 입력 받음 (n > 1)
  - 1부터 n까지의 숫자를 이용해 예시와 같은 삼각형 모양을 출력
  - 반드시 아래와 같은 함수 작성 후 여러 번 호출하여 삼각형 모양을 출력
    - **printLine(n)** : 1부터 n까지의 숫자를 한 줄로 출력 후 맨 끝에서 '줄바꿈 문자' 출력
      출력되는 각 숫자 사이에는 공백 문자가 하나씩 삽입됨
      해당 함수를 한 번 호출 시, 단 한 줄의 출력 결과만 나와야 함
  - 제출 파일: p1.py

(실행 예 1)

```
3⏎
1
1 2
1 2 3
1 2 3
1 2
1
```

(실행 예 2)

```
5⏎
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

과제 4-2

- 2번 문항
  - 아래에 제시된 함수를 구현하고 두 개의 정수를 입력 받음 (1 < n1,n2)
  - 두 개의 정수에 대한 약수를 구하여 출력
    - **problemDescription()** : 아래 문구를 출력

      ```
      * Number of divisors *
      ```

    - **getNumOfDivisors(number)** : 인자로 받은 number의 약수의 개수를 반환
      - Ex) num1 = getNumOfDivisors(12) => num1 = 6
      - Ex) num2 = getNumOfDivisors(6) => num2 = 4
  - 제출 파일: p2.py

(실행 예 1)

```
* Number of divisors *
Type the first number:
12↵
Type the second number:
6↵
Number of divisors of the first number is 6
Number of divisors of the second number is 4
```

(실행 예 2)

```
* Number of divisors *
Type the first number:
5↵
Type the second number:
18↵
Number of divisors of the first number is 2
Number of divisors of the second number is 6
```

과제 4-2

- 3번 문항
  - 정수 값을 하나 입력 받음 (1 ≤ n)
  - 실행 예와 같이 다이아몬드 모양으로 별(*)을 출력
  - 반드시 아래에 제시된 함수를 작성하여 호출
    - **printStarDia(n):** 인자로 넘겨받은 n을 이용해 다이아몬드 형태의 높이는 n*2 줄, 다이아몬드 중간의 가장 두꺼운 부분에는 별(*)이 n개 들어가는 다이아몬드 모양을 출력
    - printStarDia 함수 안에서 반드시 중첩된 반복문(for 혹은 while)을 사용
  - <<Hint>>
    - print('~~', end = '') : print() 함수 호출 시 줄 바꿈 하지 않음
    - 각 줄의 별(*)을 출력하기 전에 몇 개의 공백문자를 출력하면 되는지 생각해볼 것.
  - 제출 파일: p3.py

(실행 예 1)

```
5⏎
    *
   * *
  * * *
 * * * *
* * * * *
* * * * *
 * * * *
  * * *
   * *
    *
```

(실행 예 2)

```
4⏎
   *
  * *
 * * *
* * * *
* * * *
 * * *
  * *
   *
```

수고하셨습니다.