실전 프로그래밍

1-7-1 구구단 프로그램을 만들어보자

In [ ]:

```
for i in range(1, 10):
    for j in range(1, 10):
        print("{} x {} = {}".format(i, j, i*j))
    print()
```

```
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9

2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18

3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27

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

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45

6 x 1 = 6
6 x 2 = 12
6 x 3 = 18
```

```
6 x 4 = 24
6 x 5 = 30
6 x 6 = 36
6 x 7 = 42
6 x 8 = 48
6 x 9 = 54

7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63

8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72

9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
```

1-7-2 1~1000까지의 3의 배수의 개수, 5의 배수의 개수 구하기

In [ ]:

```
result1 = 0
result2 = 0
for n in range(1, 1000):
    if n % 3 ==0:
        result1 += 1
    if n % 5 ==0:
        result2 += 1
print(result1, result2)
```

```
333 199
```

[실습] 1~1000까지의 5의 배수의 누적합 구하기, 7의 배수의 누적합 구하기

In [ ]:

```
result1 = 0
result2 = 0
for n in range(1, 1000):
    if n % 5 ==0:
        result1 += n
    if n % 7 ==0:
        result2 += n
print(result1, result2)
```

99500 71071

1-7-3 게시물의 페이지 확인

- 게시물(m)건과 1페이지의 표시할 게시물 수(n)을 입력했을 때,
- 총 몇페이지가 될까? 계산하는 프로그램을 작성해보자.

In [ ]:

```
def NumPage(m, n):
    return m // n+1
```

In [ ]:

```
print(NumPage(5, 10), "페이지")   # 5건, 표시할 건수 10
print(NumPage(15, 10), "페이지")  # 15건, 표시할 건수 10
print(NumPage(30, 20), "페이지")  # 30건, 표시할 건수 20
```

1 페이지
2 페이지
2 페이지

[실습] 기능추가를 해 보기

- m(건수), n(표시할 건수)를 입력받는다.
- 페이지 수와 마지막 페이지에 표시될 건수를 확인하는 프로그램을 만들어보자.

In [ ]:

```
def NumPage(m, n):
    page = m // n+1
    div_sp = m % n
    return page, div_sp

m1 = int(input("총 게시물 수 : "))
n1 = int(input("한 페이지 표시 건수 :"))

page, div_sp = NumPage(m1, n1)
print("총 페이지 : {}, 마지막 페이지 게시물 건수 : {}".format(page, div_sp) )
```

총 게시물 수 : 55
한 페이지 표시 건수 :10
총 페이지 : 6, 마지막 페이지 게시물 건수 : 5

1-7-4 메모장을 만들어보자.

- 파일을 열고, 메모를 작성하기
- 파일을 열고, 현재 파일의 내용 확인해 보기

메모 작성

```
C:> python memo.py -a "Life is too short"
C:> python memo.py -a "You need pytho"
```

읽기 모드일 때, 파일 읽기

- -a : 추가 모드, -r : 읽기 모드
- -r의 경우는 memo.txt를 파일을 읽어, 이를 출력해 준다.

[실습] 새로운 파일로 메모를 해 보자.

- -w의 옵션을 이용하여 파일을 새롭게 만들수 있다

```
import sys

option = sys.argv[1]

if option=='-a':
    memo = sys.argv[2]
    f = open('memo.txt', 'a')
    f.write(memo)
    f.write('₩n')
    f.close()
elif option=='-r':
    f= open('memo.txt')
    memo = f.read()
    f.close()
    print(memo)
elif option=='-w':
    memo = sys.argv[2]
    f = open('memo.txt', 'w')
    f.write(memo)
    f.write('₩n')
    f.close()
```

```
(base) C:₩Users₩user₩Documents₩...>python part01_7_memo_03.py -w "Life is good"
```

```
(base) C:₩Users₩user₩Documents₩...>python part01_7_memo_02.py -r
Life is good
```

1-7-5 하위 디렉터리 검색해 보기

In [ ]:

```
import os

for (path, dir, files) in os.walk("C:/Users/user/Documents/doit"):
    for filename in files:
        ext = os.path.splitext(filename)[-1]
        if ext=='.py':
            print("%s%s" % (path, filename))
```

```
C:/Users/user/Documents/doitmod1.py
C:/Users/user/Documents/doitmod2.py
C:/Users/user/Documents/doitmodtest.py
C:/Users/user/Documents/doitseaborn.py
```

1-7-6 정규 표현식

- 정규 표현식(Regular Expressions)은 복잡한 문자열을 처리할 때, 사용하는 기법. 파이썬만의 고유 문법이 아니라 문자열을 처리하는 모든 곳에서 사용한다.

정보의 암호화 처리

- 우리는 숫자-숫자 에서 뒤의 숫자를 암호화를 처리하여 안보이도록 해야 한다.

In [ ]:

```
data = """
park,10234-1422351
lim ,22342-1422251
"""
```

In [ ]:

```
result = []
for line in data.split("₩n"):  # 한줄 단위로 구분
    word_result = []

    one_line = line.split(",")  # 한줄 데이터를 공백으로 나누기

    for word in one_line:
        if len(word) == 13 and word[:5].isdigit() and word[6:].isdigit():
            word = word[:6] + "-" + "*******"
        word_result.append(word)
    result.append(" ".join(word_result))
print("₩n".join(result))
```

```
park 10234--*******
lim  22342--*******
```

In [ ]:

```
import re

data = """
park 80112-1422351
lim  81012-1422251
"""

pat = re.compile("(₩d{5})[-]₩d{7}")  # 숫자 5개, 숫자 7개
print(pat.sub("₩g<1>-*******", data))
```

```
park 80112-*******
lim  81012-*******
```