

실전 프로그래밍

01 구구단 프로그램 만들기

In [1]:



```
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

02. 게시물의 페이지 확인 계산 프로그램

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

In [2]:



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

In [3]:



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

1 페이지

In [4]:



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

2 페이지
2 페이지

[실습] 추가 기능

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

In [5]:



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

In [6]:



```
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 : 100
한 페이지 표시 건수 : 7
총 페이지 : 15, 마지막 페이지 게시물 건수 : 2

03 하위 디렉터리 검색 후, 원하는 파일 형태만 출력

- html 파일만 출력

os.walk(대상경로) : 하위의 폴더들을 반복적으로 탐색

- 인자로 전달된 경로에 대해 3개의 값의 tuple로 넘겨준다.
 - path : dir과 files가 있는 경로
 - dirs : path아래에 있는 폴더들
 - files : root아래에 있는 파일들
- os.path.splitext(filename) : 파일의 확장자를 알아보기
 - 예제 결과 : ('D:\Github\CLASS_PYTHON_START01_python_start', '.html')
 - 두번째 값이 파일의 확장자

In [12]:



```
import os

for (path, dir, files) in os.walk("D:\\Github\\CLASS_PYTHON_START"):
    for filename in files:
        ext = os.path.splitext(filename)[-1]
        if ext == '.html':
            print("%s%s" % (path, filename))
```

```
D:\\Github\\CLASS_PYTHON_START\\01_python_start.html
D:\\Github\\CLASS_PYTHON_START\\02_python_start_if_for_list.html
D:\\Github\\CLASS_PYTHON_START\\03_python_start_file.html
D:\\Github\\CLASS_PYTHON_START\\04_python_start__fnc_module.html
D:\\Github\\CLASS_PYTHON_START\\05_Seaborn_Basic.html
D:\\Github\\CLASS_PYTHON_START\\06_ml_start.html
```

04 정보의 암호화 하기 - 일반적 방법

In [13]:



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

In [14]:



```
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-*****
```

04 정보의 암호화 하기 - 정규 표현식

In [16]:



```
import re

data = """
park 80112-1422351
lim 80102-1422251
"""

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

```
park 80112-*****
lim 80102-*****
```

05 예외처리 exception 알아보기

5-1 문자로 나눌 때, 이에 대한 예러 메시지 출력하기

In [17]:



```
def divide(a, b):
    return a/b

try:
    c = divide(5, 'string')
except ZeroDivisionError:
    print("두번째 인자는 0이어서는 안됩니다.")
except TypeError:
    print("모든 인자는 숫자여야 합니다.")
except:
    print("무슨 에러?")
```

모든 인자는 숫자여야 합니다.

In [20]:



```
def divide(a, b):
    return a/b

try:
    c = divide(5, 'kkk')
except ZeroDivisionError:
    print("두번째 인자는 0이어서는 안됩니다.")
except TypeError:
    print("모든 인자는 숫자여야 합니다.")
except Exception:
    print("무슨 에러?")
```

모든 인자는 숫자여야 합니다.

- (실습해보기) 나누는 값을 0으로 할 때는 어떻게 되는가?

5-2 예외처리 try, except, else, finally

```
try:
    실행할 코드
except:
    예외가 발생했을 때 처리하는 코드
else:
    예외가 발생하지 않았을 때 실행할 코드
finally:
    예외 발생 여부와 상관없이 항상 실행할 코드
```

In [22]:



```
def divide(a, b):
    return a/b

try:
    c = divide(5, 1)
except ZeroDivisionError:
    print("두번째 인자는 0이어서는 안됩니다.")
except TypeError:
    print("모든 인자는 숫자여야 합니다.")
else:
    print("예외 발생 없이 정상 실행")
finally:
    print("모든 프로그램 정상 실행하였음")
```

예외 발생 없이 정상 실행
모든 프로그램 정상 실행하였음

5-3 예외 발생시키기

In [24]:



```
try:
    x = int(input('5의 배수를 입력하세요: '))
    if x % 5 != 0:
        raise Exception('5의 배수가 아닙니다.')
    print(x)
except Exception as e:
    print('예외가 발생했습니다.', e)
```

x가 5의 배수가 아니면
예외를 발생시킴
예외가 발생했을 때 실행됨

5의 배수를 입력하세요: 3
예외가 발생했습니다. 5의 배수가 아닙니다.

숫자를 입력하지 않았을때,

- `except Exception as e` 구문을 이용하여 에러 메시지 출력 가능

In [27]:



```
try:
    x = int(input('5의 배수를 입력하세요: '))
    print(x)
except Exception as e:
    print('예외가 발생했습니다.', e) # 예외가 발생했을 때 실행됨
```

5의 배수를 입력하세요: kk
예외가 발생했습니다. invalid literal for int() with base 10: 'kk'

기타

In [2]:



```
try:
    3 / 0
except IndexError as e:
    print("인덱스가 안 맞습니다.", e)
except ZeroDivisionError as e:
    print("0으로 나누면 안됩니다.", e)

print("프로그램의 정상 종료!")
```

0으로 나누면 안됩니다. division by zero
프로그램의 정상 종료!

else절은 블록에서 예외가 발생하지 않았을 경우에 실행. 생략 가능

In [4]:



```
try:
    pass
except IndexError as e:
    print("인덱스가 안 맞습니다.", e)
except ZeroDivisionError as e:
    print("0으로 나누면 안됩니다.", e)
else:
    print("예외가 발생 안됨!")

print("프로그램이 정상적으로 종료됩니다!")
```

예외가 발생 안됨!
프로그램이 정상적으로 종료됩니다!

finally 절은 예외와 상관없이 언제나 실행

In [5]:



```
try:
    3 / 0
except IndexError as e:
    print("인덱스가 안 맞습니다.", e)
except ZeroDivisionError as e:
    print("0으로 나누면 안됩니다.", e)
finally:
    print("예외에 상관없이 언제나 실행!")

print("프로그램이 정상적으로 종료됩니다!")
```

0으로 나누면 안됩니다. division by zero
예외에 상관없이 언제나 실행!
프로그램이 정상적으로 종료됩니다!

In []:

