예외 처리

try ~ except

형식1

```
try:
    실행할 코드
except:
    예외가 발생했을 때 처리하는 코드
```

- except에서 모든 예외처리를 하는 형식

형식2

```
try:
    실행할 코드
except 예외명:
    예외가 발생했을 때 처리하는 코드
```

- 특정예외만 처리하는 형식
    - ZeroDivisionError : 숫자를 0으로 나눠서 발생하는 에러
    - IndexError : 범위를 벗어난 인덱스에 접근하여 발생하는 에러
    - AttributeError : 모듈, 클래스의 잘못된 속성 사용시 발생하는 에러
    - NameError : 변수선언 없이 변수를 사용하여 발생하는 에러

형식3

```
try:
    실행할 코드
except 예외명 as 변수:
    예외가 발생했을 때 처리하는 코드
```

- 예외의 에러 메시지를 변수로 받아오는 형식
- 변수명은 주로 except의 첫글자인 e를 사용한다.

else와 finally

형식

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

- else는 except 바로 다음에 와야 하며, except를 생략할 수 없다.
- finally는 예외 발생 여부와 상관없이 항상 코드를 실행하며, except와 else를 생략할 수 있다.

임의로 예외 발생시키기

숫자를 0으로 나눴을때나 리스트의 인덱스를 벗어난 경우와 같이 파이썬에서 이미 정의된 예외 이외에 개발자가 직접 예외를 발생시킬 수 있다.

형식

```
try:
    if 예외조건:                          # if문으로 예외 판단
        raise Exception('예외메세지')    # 예외를 발생시킴
except Exception as e:
    print('예외가 발생했습니다.', e)     # 위에서 정한 예외메세지가 출력됨
```

- 개발자가 직접 예외를 발생시킬 때는 raise에 예외를 지정하고 에러 메시지를 정의한다.

예외클래스 만들기(사용자 정의 예외)

파이썬에서 이미 정의된 예외이외에 개발자가 직접 예외클래스를 정의할 수 있다.

형식

```
class 예외이름(Exception):
    def __init__(self):
        super().__init__('에러메시지')
```

- Exception을 상속받아서 새로운 클래스를 만든다.
- __init__ 메서드에서 부모클래스의 __init__ 메서드를 호출하면서 에러 메시지를 정의한다.

예제] 16tryexcept.py

```
def calc(val):
    sum = None
    try:
        sum = val[0] + val[1] + val[2]
        if val[0]==100:
            print(no_var)
        elif val[0]==55:
            result = val[0] / 0
            print("결과", result)
    except IndexError:
        print('리스트의 인덱스에 에러가 있습니다')
    except NameError as err:
        print('선언되지 않은 변수를 사용하였습니다', str(err))
    except:
        print('예외가 발생하였습니다')
    else:
        print('에러없음^^')
    finally:
        print('sum', sum)

```

```
print('실행1')
calc([1, 2, 3])
print('실행2')
calc([10, 20])
print('실행3')
calc([100,101,102,103])
print('실행4')
calc([55,56,57])
```

```
print('실행5')
try:
    fp = open("test.txt", "r")
    try:
        lines = fp.readlines()
        print(lines)
    finally:
        print("파일 객체를 닫습니다")
        fp.close()
except IOError:
    print('파일에러발생')

```

```
print('실행6')
try:
    x = int(input('3의 배수를 입력하세요: '))
    if x % 3 != 0:
        raise Exception('[예외메세지] 3의 배수가 아님')
    print(x)
except Exception as e:
    print('예외가 발생했습니다.', e)

```

```
print('실행7')
class GugudanRangeExcept(Exception):
    def __init__(self):
        super().__init__('구구단의 범위를 벗어났습니다')

def print_gugudan(end_num):
    try:
        if end_num > 9:
            raise GugudanRangeExcept
        end_su = end_num + 1
        for su in range(2,end_su):
            for dan in range(1,10):
                print("%2d * %2d = %2d" % (su, dan, su*dan), end=' ')
            print()
        print()
    except Exception as e:
        print('예외발생', e)

print_gugudan(int(input("출력할 단 수를 입력하세요:")))
```