

**Fastcampus**

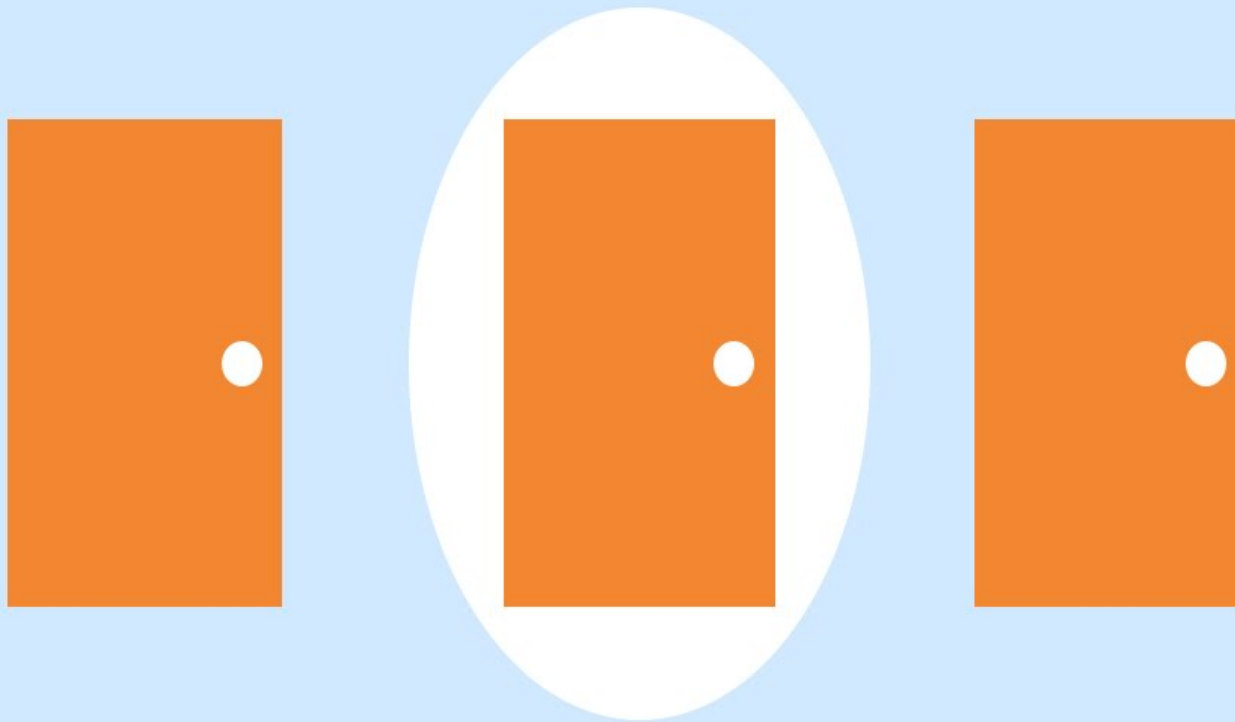
**Computer Science Extension School**

**Python Basic\_Day4**

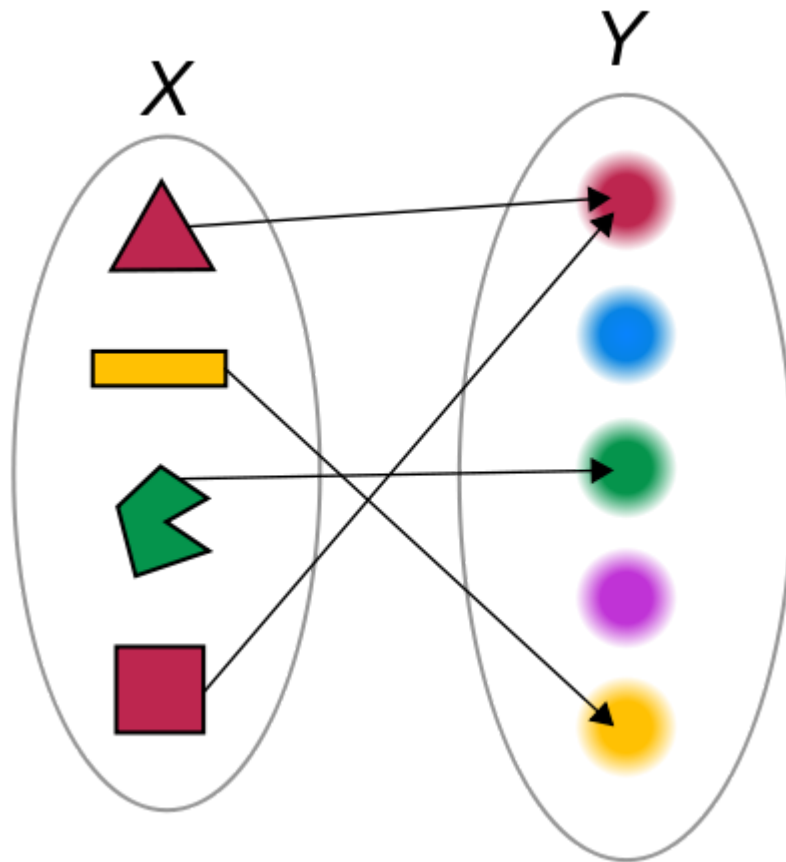


# Monty Hall Problem

# MONTY HALL PROBLEM

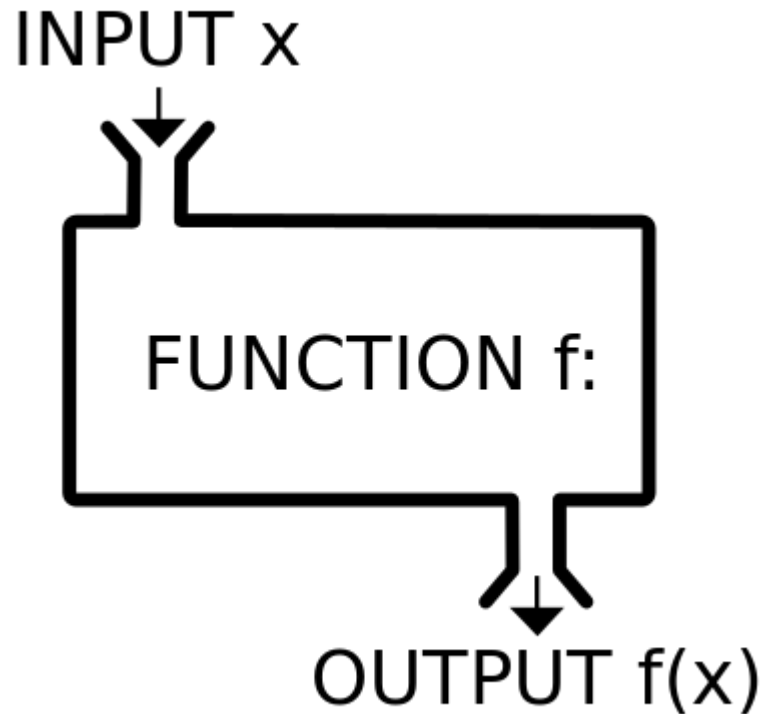


# function



- 수학적 정의: 첫 번째 집합의 임의의 한 원소를 두 번째 집합의 오직 한 원소에 대응시키는 대응 관계
- $x$ : 정의역  $y$ : 공역

# function



- 프로그래밍에서의 함수: 입력값을 내부에서 어떤 처리를 통해 결과값을 출력하는 것

# function

```
def function(parameter):  
    실행문1  
    실행문2  
    ...  
    return output
```

# function

```
def awe_sum(a,b):  
    result = a + b  
    return result  
  
a = 2  
b = 3  
print(awe_sum(a,b))
```



## function without input

```
def print_hello():  
    return "hello"  
  
result_hello = print_hello()  
print(result_hello)
```

## function without return

```
def func_wo_return(a):  
    print("This is function without return for " + str(a) + " times")  
  
func_wo_return()
```

## function with multiple return

```
def mul_return(a):  
    b = a + 1  
    return a, b
```

## return skill

```
def id_check(id):  
    if id == "admin":  
        print("invalid id: admin")  
        return  
    print("valid id: ", id)
```

## parameter with initialize

```
def say_hello(name="Fool", nick=True):  
    print("Hi, ", name)  
    if nick == True:  
        print("But, you are Fool")  
    else:  
        print("Oh, you are not Fool")
```

초기값을 설정할때 항상 그 인자를 마지막에 두어야 합니다.

## arguments

```
def mul_sum(*args):  
    sum = 0  
    for i in args:  
        sum += i  
    return sum
```

# keyword arguments

```
def show_kwargs(**kwargs):  
    print(str(kwargs))  
  
show_kwargs(a=10, b="google")
```

## keyword arguments

```
def kwargs_url(server, port, **query):  
    url = "https://" + server + ":" + port + "?"  
    for key in query.keys():  
        url += key + "=" + query[key] + "&"  
    return url  
  
kwargs_url("localhost", "8080", utm_source="google", keyword="nav
```



# variable outside function

```
a = "hello"
def glob_test(a):
    a += "world"
    return a

glob_test(a)
print(a)
```

```
a = "hello"
def glob_test(x):
    x += "world"
    return x

glob_test(a)
print(a)
```

## variable outside function

```
def glob_test2(x):  
    x += "world"  
    return x
```

```
glob_test2("hello")  
glob_test2(x)
```

# So, how to globalize

(1) using return

```
a = "hello"
def glob_test(a):
    a += "world"
    return a

a = glob_test(a)
print(a)
```

# So, how to globalize

(2) use global

```
a = "hello"
def glob_test(a):
    global a
    a += "world"
    return a

glob_test(a)
print(a)
```

global 이라는 명령을 사용하여 전역변수로 사용하게 되면 함수는 독립성을 잃게 되어 함수가 외부변수에 의존적이게 됩니다.

## Leap year

4로 나뉘어 떨어지면 윤년,  
100으로 나뉘어 떨어지면 평년,  
400으로 나뉘어 떨어질때 윤년

## Leap year(answer)

```
leap = False
def is_leap(y):
    if y % 4 == 0 and (y % 100 != 0 or y % 400 == 0):
        leap = True
    return leap

y = int(input("Is leap?? "))
print(is_leap(y))
```

## numguess with function

```
def guesser(guess):  
    if guess == answer:  
        print("Correct! The answer was ", str(answer))  
        break  
    else:  
        print("That's not what I wanted!! Try again!!")
```

# Recursive

```
times = int(input("How many times want to curse the beast??: "))
def recurse_beast(a):
    if a == 0:
        print("curse complete!")
    else:
        print("Fusion!!!(%d times left)" % a - 1)
        recurse_beast(a-1)

recurse_beast(times)
```