

# 파이썬 문법

## 1. 숫자

### 사칙 연산

- 덧셈 : +
- 뺄셈 : -
- 나누기 : /(소숫점 有) & //(소수점 無) & %(나머지 有)
- 곱셈 : \*(그냥 곱셈) & \*\*(거듭제곱)

### 숫자 형식의 종류

- int : 정수
- float : 실수(혹은 부동소수점)

## 2. 변수

### 변수 선언

- ex) x= 10
- ex) x, y, z =1, 2, 3
- ex) x=None // 빈 변수 선언

### 변수 삭제

- ex) x=10
- del x

## 3. 입력값

### 입력값

- ex) x=input("문자열을 입력하시오") // str값
- ex) x=int(input("숫자를 입력하시오") // int 값

ex) `x=float(input("숫자를 입력하시오"))` // float 값

### 두개의 입력값

ex) `x,y=input("문자열을 두개 입력하시오").split()` // 공백 기준으로 두개 분리

### 유형 변경

ex) `x = input("숫자를 입력하시오")` // 7 입력

`x = int(x)` // str 7이 int 7로 변경

`>>type : int`

## 4. 출력값

### 출력값 간격, 띄우기, 붙이기

ex) `print(1,2,3, sep=',')` // 매 step마다 sep의 값들을 삽입

`>> 1,2,3`

ex) `print(1, 2, 3, sep='\n')`

`>> 1`

`>> 2`

`>> 3`

ex) `print(1, end="")`

`print(2, end="")`

`print(3)`

`>> 123`

## 5. 비교 및 논리 연산자

### TRUE & FALSE

ex) `3>1`

```
>> True
```

```
ex) 10 == 5
```

```
>> FALSE
```

### **AND & OR**

```
ex) TRUE and TRUE
```

```
>> TRUE
```

```
TRUE and FALSE
```

```
>> FALSE
```

```
>> AND : 둘 중에 둘다 만족해야 TRUE
```

```
ex) TRUE and TRUE
```

```
>> TRUE
```

```
TRUE and FALSE
```

```
>> TRUE
```

```
FALSE and FALSE
```

```
>> FALSE
```

```
>> OR : 둘 중에 하나라도 만족하면 TRUE
```

## **6. 리스트 list & 튜플 tuple**

**리스트 : 내용 변경 가능**

```
ex) 리스트 = []
```

ex) 리스트 = list()

### 튜플 : 내용 변경 불가능

ex) a = ( 1, 2, 3, 4, 5)

ex) a= 1, 2, 3, 4, 5

## 7. 인덱스 index

### 인덱스

ex) a= [10, 11, 12 ,13 ,14, 15]

a[0]

>> 10 // 리스트의 첫번째 인덱스 요소 출력

a[-1]

>> 15 // 리스트의 뒤에서 첫번째 인덱스 요소 출력

ex) a= (10, 11, 12, 13, 14 ,15)

a[0]

>> 10 // 튜플의 첫번째 인덱스 요소 출력

a[-1]

>> 15 // 튜플의 뒤에서 첫번째 인덱스 요소 출력

# len()

ex) a= [10, 11, 12, 13, 14, 15]

len(a)

>> 6 // 리스트 a의 길이

# del ??

ex) a= [10, 11, 12, 13, 14]

del a[2] // 리스트의 세번째 인덱스 요소 제거

print(a)

>> [10, 11, 13, 14]

## 7. 슬라이스

### 슬라이스

ex) a= [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

a[1:1]

>> [] // 인덱스 1 ~ 1까지 자른다 >> 아무것도 없다

a[1:2]

>> [2] // 인덱스 1 ~ 2까지 자른다 >> 인덱스 1

>> 왼쪽값 포함, 오른쪽값 미포함

a[0:6:2]

>> [1, 3, 5] // 인덱스 0부터 5까지 2씩 증가시키면서 가져옴

a[:3]

>> [1,2,3,4] // 처음부터 인덱스 3까지 가져옴

a[7:]

>> [8, 9 ,10] // 인덱스 7부터 끝까지 가져옴

a[:]

>> [1,2,3,4,5,6,7,8,9,10] // 인덱스 전체를 가져옴

## 8. 조건문

### if & else

if 조건식 :

코드

ex) x= 10

```
if x== 10:
    print("x는 10입니다.")
>> x는 10입니다.
```

```
if 조건식 :
    코드
else :
    코드
```

```
ex ) x= 10
if x== 10:
    print("X는 10 입니다.")
else :
    print("X는 10이 아닙니다.")
>> X는 10 입니다.
```

### elif

```
if 조건식 :
    코드
elif:
    코드
```

```
ex) x= 20
if x==10:
    print("10입니다.")
elif x== 20:
    print("20입니다.")
>> 20입니다.
```

## 9. 반복문

### for & range

for 변수 in range(횟수) :

    코드

ex) sum=0

    for i in range(100):

        sum+=i

    print(sum)

>> 5050 // 0 ~ 100까지 숫자들의 합

    for i in range(10,0,-1):

        print(i)

>> 10

>> 9

>> ...

>> 1 // 10에서 1까지 1씩 감소

    for i in reversed(range(10)):

        print(i)

>> 9

>> 8

>> ...

>> 0 // 9에서 0까지 10번 반복

### while

    초기식

while 조건식 :

    반복할 코드

## 변화식

ex) i=0

```
while i < 100 :
```

```
    print("*")
```

```
    i+=1
```

```
>> *
```

```
>> *
```

```
>> ...
```

```
>> * // 0번부터 99번째까지 총 100번 출력
```

ex)

```
while True :
```

```
    print('Hello!')
```

```
>> Hello!
```

```
>> ... // 무한 루프
```

## break

ex) i=0

```
while True:
```

```
    i+=1
```

```
    if i == 100:
```

```
        break // 반복문을 끝낸다.
```

ex)

```
for i in range(10000):
```

```
    if i == 100:
```

```
        break // 반복문을 끝낸다
```



## continue

ex)

```
for i in range(100):
```

```
    if i % 4 == 0:
```

```
        continue // continue가 실행될 시 아래의 코드는 뛰어넘는다
```

```
    print(i)
```

## 10. 함수

### 함수

def 함수 이름 :

코드

```
def hello():
    print('Hello, World!')

hello()
```

```
def add(a,b):
    return a+b

x= add(10,20)
>> x
30
```

return : 결과값 반환, 항상 끝에 위치하지 않더라도 반환시킬 수 있다.

```
def add(a,b):
    return a+b, a-b

x,y=add(10,2)
>> x
```

```
12
>> y
8
```

결과값을 여러개로 반환시킬 수 있다.

## 위치인수

```
def print_number(a,b,c):
    print(a)
    print(b)
    print(c)

x=[10,20,30]
print_number(*x)
>>10
20
30

>>> print_number(*[10, 20, 30])
10
20
30
```

언패킹(unpacking) : 리스트를 푸는 것

```
def print_number(*args):
    for arg in args:
        print(arg)

x=[10]
y=[10,20,30,40]

>>> print_number(*x)
10

>>> print_number(*y)
10
20
30
40
```

## 키워드 인수

```
def personal(name, age, address):
    print('이름 :', name)
    print('나이 :', age)
    print('주소 :', address)

x = {'name': '이원목', 'age' : 25, 'address' : '대구'}
>>> personal(**x)
이름 : 이원목
나이 : 25
주소 : 대구
```

**\*\*** 붙이는 이유 : 딕셔너리는 두번 언패킹이 진행되기 때문

## 11. 클래스

### 클래스

```
class Name :
    def function(self):
        print('Hello')

james=Name() // james가 Name의 인스턴스
james.function
>> Hello
```

```
class Name:
    def __init__(self):
        self.hello="Hello"

    def function(self)
        print(self.hello)

james=Name()
james.function
>> Hello
```

**init** : 인스턴스(객체)를 초기화

**self** : 인스턴스 자기 자신

james에 해당하는 부분에 self라는 매개변수가 자동으로 입력

```
class Name:
    def __init__(self, name, age, address):
        self.hello = 'Hello'
        self.name = name
        self.age = age
```

```

        self.address = address

    def function(self):
        print('{0} 저는 {1}입니다.'.format(self.hello,self.name))

Lee = Name('LeewM',25, '대구')
Lee.function

print('이름:',Lee.name)
print('나이:',Lee.age)
print('주소:',Lee.address)
>> Hello 저는 LeewM입니다.
이름 : LeewM
나이 : 25
주소 : 대구

```

init : self 다음에 이름, 나이, 주소 순으로 지정

print('{0},{1},{2}'.format(a,b,c)) : 출력값에 지정 부분에 값 입력 가능

```

class Name:
    def __init__(self, name, age, address, wallet):
        self.name = name
        self.age =age
        self.address = address
        self.__wallet = wallet // 변수 앞에 __ 를 붙여 비공개로

Lee=Name('LeewM',25, '대구',5000)
>>>Lee.__wallet+=10000
Error // 비공개 속성에 접근할 시 에러 발생

```

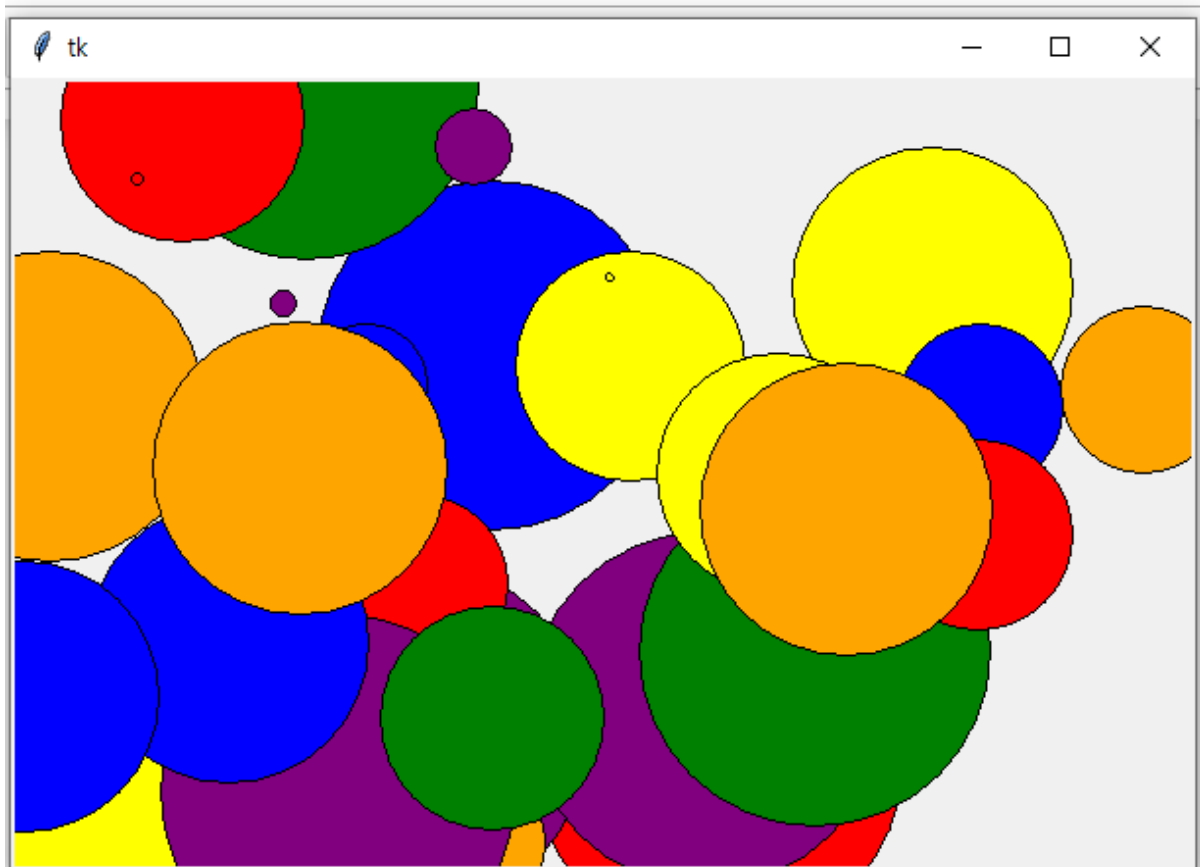
# 파이썬 tkinter

## 원 그리기

```
import tkinter as tk
import random as rn
```

```
win=tk.Tk()
W=600; H=400
cnv=tk.Canvas(win,width=W,height=H)
cnv.pack()

colors=["red","purple","blue","green","yellow","orange"]
for i in range(30):
    x=W*rn.random();y=H*rn.random()
    r=100*rn.random()
    cnv.create_oval(x-r,y-r,x+r,y+r,fill=colors[i%6])
win.mainloop()
```



## tkinter 설치

윈도우 창 >> cmd >> pip install tkinter

필요시 python -m pip install --upgrade pip ( pip 업그레이드 필요)

# 파이썬 wxPython

## wxPython 설치

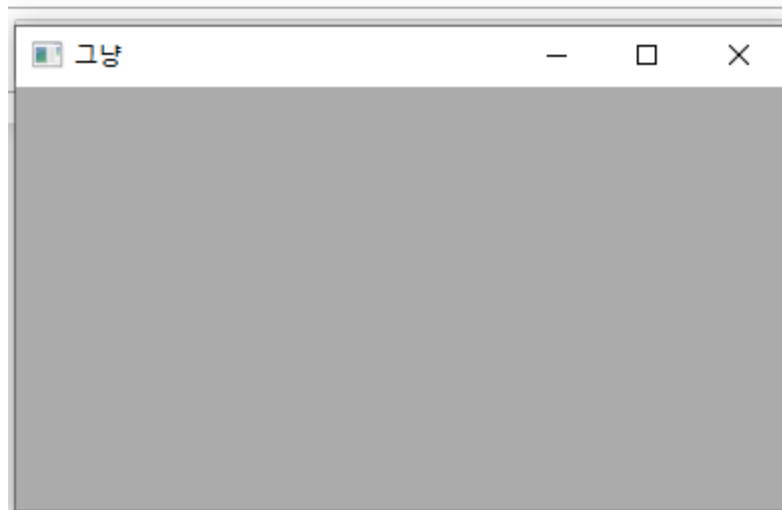
윈도우 창 >> cmd >> pip install wxPython

필요시 python -m pip install --upgrade pip ( pip 업그레이드 필요)

```
import wx
import random as ran
```

```
app = wx.App()
frame = wx.Frame(None, -1, '그냥')
frame.Show()

app.MainLoop()
```

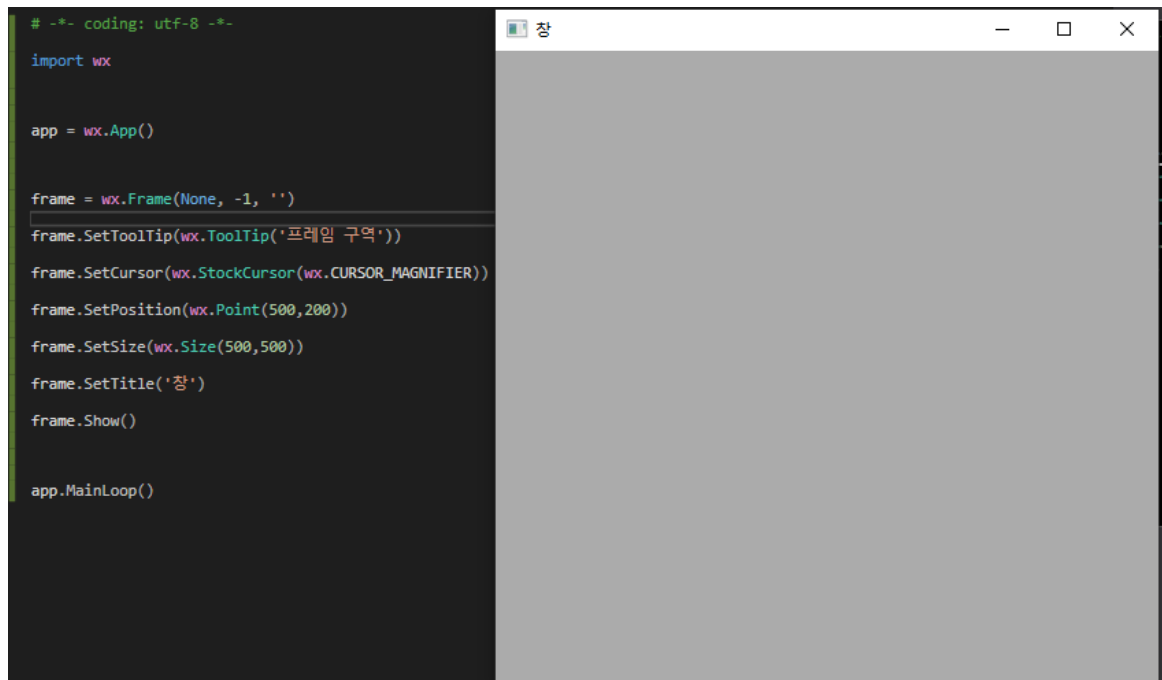


app = wx.App() : 어플리케이션 오브젝트 생성

frame = wx.Frame(None, -1, '타이틀 명') : 실제 화면인 프레임 생성

frame.Show() : 생성한 프레임을 실제로 보여줌

app.MainLoop() : 어플리케이션이 계속해서 실행하면서 발생하는 이벤트 받아들이



SetToolTip : 커서의 정지상태일 때의 메세지 창

SetCursor : 커서의 상태 지정

SetPosition : frame의 위치 지정

SetSize : frame의 크기 지정

SetTitle : frame의 이름 지정