

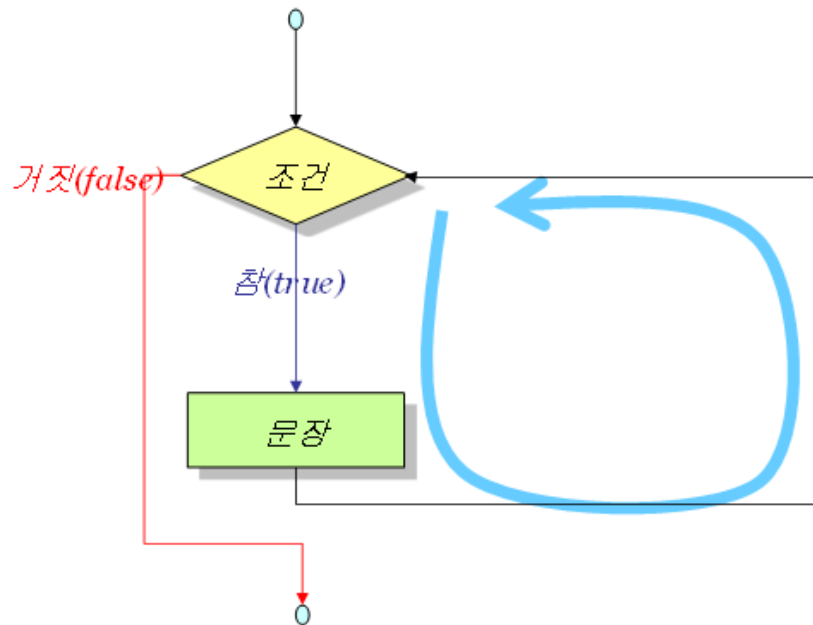
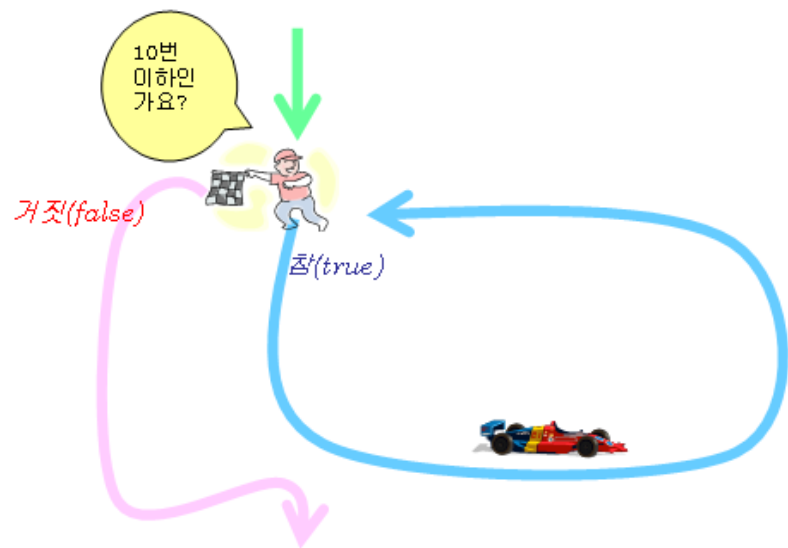


단국대학교
SW중심대학

창의적 사고와 코딩

Lecture 4-2. 반복 (while)

- ❖ while 문은 조건을 정해놓고 반복을 하는 구조이다.



while 문의 구조

전체적인 구조



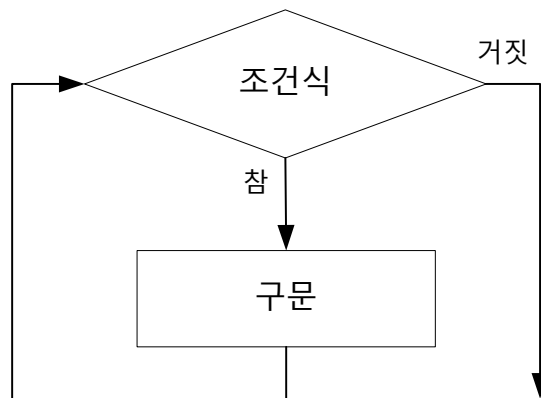
while **조건** :

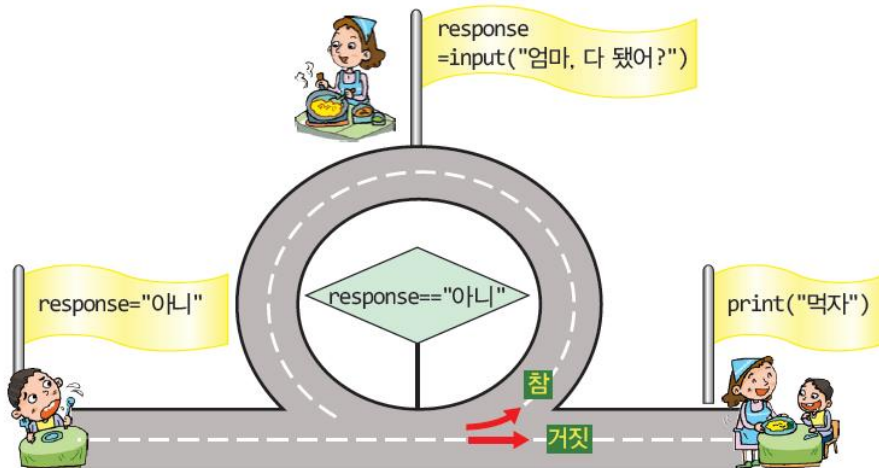
반복 문장

반복 문장

반복을 하는 조건이다. 조건이 참이면 반복을 계속한다.

반복되는 문장이다.





```
response = "아니"
while response == "아니":
    response = input("엄마, 다 됐어?");
print("먹자")
```

- ❖ 사용자가 암호를 입력하고 프로그램에서 암호가 맞는지를 체크한다.

```
암호를 입력하시오: hello  
암호를 입력하시오: idontknow  
암호를 입력하시오: 12345678  
암호를 입력하시오: pythonisfun  
로그인 성공
```

```
password = ""  
while password != "pythonisfun":  
    password = input("암호를 입력하시오: ")  
print("로그인 성공")
```

예제 compare_for_while

```
i = 0
while i < 5:
    print ("환영합니다.")
    i = i + 1
print("반복이 종료되었습니다.")
```

환영합니다.
환영합니다.
환영합니다.
환영합니다.
환영합니다.
반복이 종료되었습니다.

```
for i in range(5): # range(0,5,1)
    print("환영합니다.")
print("반복이 종료되었습니다.")
```

- ✧ 0, 1, 2, ..., 9까지를 차례대로 화면에 출력하는 프로그램을 작성하여 보자.
 - 변수 i 의 값을 0으로 초기화하고 반복하면서 i 를 출력하고 1씩 증가시키면 된다.
 - i 가 10보다 작을 때까지 반복시키면 된다.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|

Solution : 파일명 while1.py



Lab: $(1+2+3+\dots+9+10)$ 계산하기

- ✧ $(1+2+3+\dots+9+10)$ 의 값을 계산하는 프로그램을 작성하여 보자. 이것은 공식으로도 계산할 수 있으나 우리는 반복 구조를 사용해보자.

합계 = 55



1

1



1+2

= 3



1+2+3

= 6



1+2+3+4

= 10

Solution : 파일명 while2.py



- ✧ 팩토리얼을 계산하는 프로그램을 작성하여 보자. 팩토리얼 $n!$ 은 1부터 n 까지의 정수를 모두 곱한 것을 의미한다. 즉, $n! = 1 \times 2 \times 3 \times \dots \times (n-1) \times n$ 이다. 예를 들어서 $10!$ 을 계산하는 프로그램을 작성하여 보자.

팩토리얼을 구할 정수: 10
10!은 3628800입니다.

Solution : 파일명 facto.py



- ✧ 구구단 중에서 3단을 반복문을 이용하여 출력하여 보자. $3*1$, $3*2$, $3*3$, ..., $3*9$ 까지 9번 반복시키면 출력하면 될 것이다.

원하는 단은: 5

$$5*1=5$$

$$5*2=10$$

$$5*3=15$$

$$5*4=20$$

$$5*5=25$$

$$5*6=30$$

$$5*7=35$$

$$5*8=40$$

$$5*9=45$$

```
dan = int(input("원하는 단은: "))  
i = 1  
  
while i <= 9:  
    print("%s*%s=%s" % (dan, i, dan*i))  
    i = i + 1
```

```
#for  
dan = int(input("원하는 단은: "))  
  
for i in range(1, 10):  
    print("%s*%s=%s" % (dan, i, dan*i))
```

5가 입력 될 때 까지 임의의 수 선택하기

- ✦ 0부터 9까지의 무작위 숫자를 반복적으로 선택하여 출력하다가 5가 선택되면 5를 출력 후 종료된다.

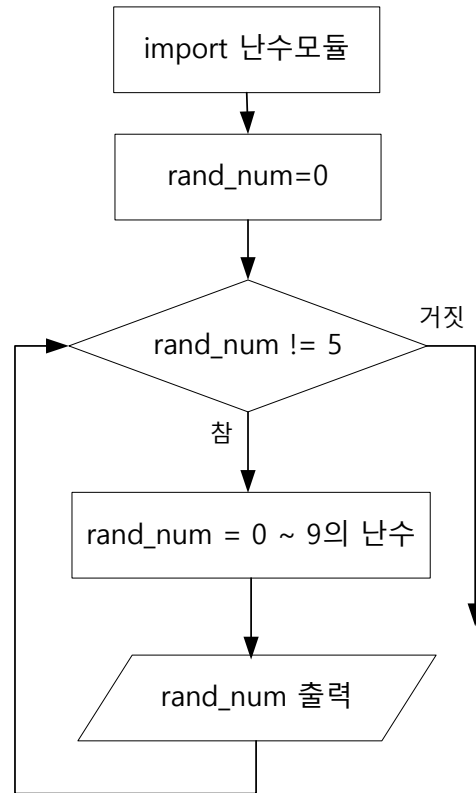
```
7  
2  
0  
1  
5  
>>>
```

Solution : 파일명 num_select.py

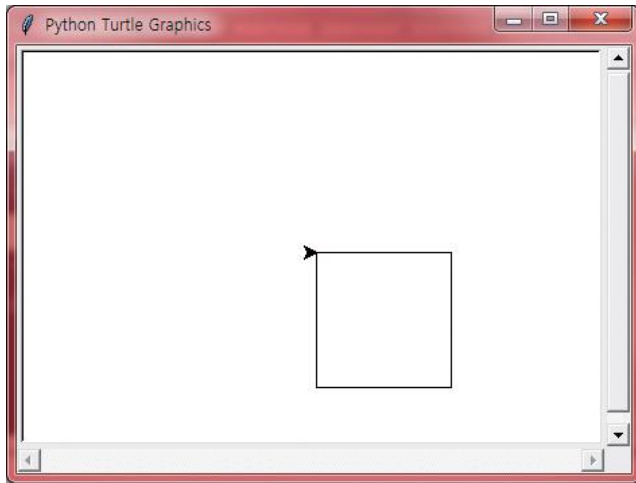
```
import random

rand_num = 0

while rand_num != 5:
    rand_num = random.randint(0, 9)
    print(rand_num)
```



- ✧ **while** 루프를 이용하여서 화면에 사각형을 그리는 코드를 작성해보자.



```
import turtle
t = turtle.Turtle()
i = 0
while i < 4:
    t.forward(100)
    t.right(90)
    i = i + 1
```

```
#for문
import turtle
t = turtle.Turtle()

for i in range(4):
    t.forward(100)
    t.right(90)
```

- ✧ 1000만원을 은행에 저금한다고 가정하자. 현재 이율은 5%로 몇 년이 지나야 원금의 두 배가 될까?
 - 변수: year(기간), balance(금액), interest(이율)

원금의 두배가 되는 기간(년): 15
총액: 2078.928179411367

Solution : 파일명 invest.py

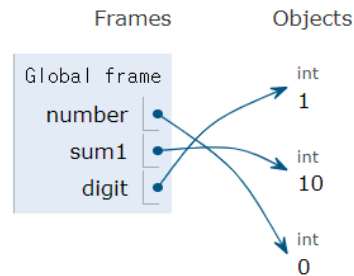
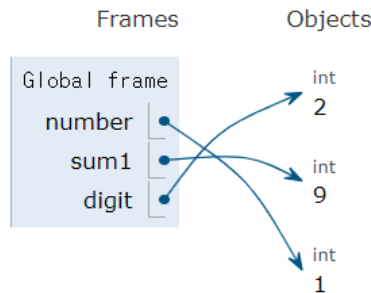
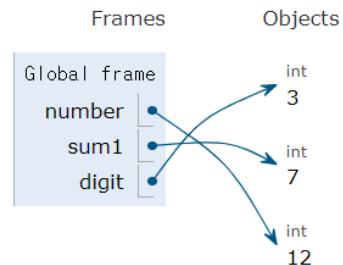
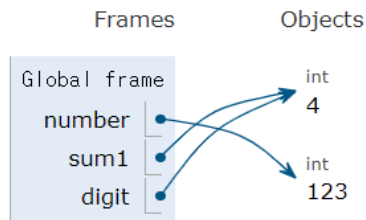
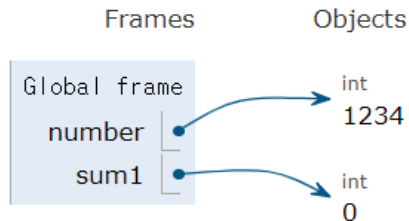


- ✧ 정수 안의 각 자리수의 합을 계산하는 프로그램을 작성해보자. 예를 들어서 1234라면 $(1+2+3+4)$ 를 계산하는 것이다.
 - 변수: number(1234), sum1(자리수 합), digit(일의 자리)
 - 반복 조건: number > 0

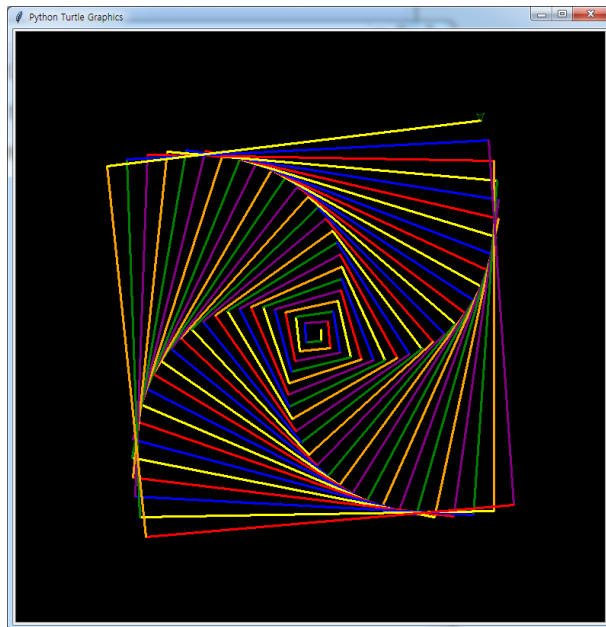
자리수의 합은 10입니다.

Solution

```
number = 1234
sum1 = 0
while number > 0 :
    digit = number % 10
    sum1 = sum1 + digit
    number = number // 10
print("자리수의 합은 %d입니다." % sum1)
```



- ❖ 사각형을 그리는 것이지만 한 번 반복할 때마다 각도가 90도가 아니라 89도로 회전
 - 색상은 리스트에 저장했다가 하나씩 꺼내 변경(그릴때마다 선택이 순서대로 바뀌도록)
`colors = ["red", "purple", "blue", "green", "yellow", "orange"]`
 - 배경색 변경
 - 거북이의 속도 설정 (0이 최대 속도)
 - 거북이가 그리는 선의 두께



Solution : 파일명 Spiral.py



```
import turtle

# 색상은 리스트에 저장했다가 하나씩 꺼내서 변경하도록 하자.
colors = ["red", "purple", "blue", "green", "yellow", "orange"]
t = turtle.Turtle()

# 배경색은 다음과 같은 문장으로 변경이 가능하다.
turtle.bgcolor("black")

# 거북이의 속도는 0으로 설정하면 최대가 된다.
t.speed(0)

# 거북이가 그리는 선의 두께는 width()를 호출하면 된다.
t.width(3)

length = 10      # 초기 선의 길이는 10으로 한다.

# while 반복문이다. 선의 길이가 500보다 작으면 반복한다.
while length < 500:
    t.forward(length)          # length만큼 전진한다.
    t.pencolor(colors[length%6]) # 선의 색상을 변경한다.
    t.right(89)                # 89도 오른쪽으로 회전한다.
    length += 5                # 선의 길이를 5만큼 증가한다.
```


Lab 사용자가 입력하는 숫자의 합 계산하기



- ❖ 사용자가 입력하는 숫자를 더하는 프로그램
- ❖ 사용자가 yes라고 답한 동안에만 숫자를 입력 받음

```
숫자를 입력하시오: 10  
계속?(yes/no): yes  
숫자를 입력하시오: 20  
계속?(yes/no): yes  
숫자를 입력하시오: 5  
계속?(yes/no): no  
합계는 : 35
```

- total을 0으로 설정
- answer를 'yes'로 설정
- answer가 'yes'인 동안에 다음을 반복
 - ① 숫자를 입력 받는다
 - ② 숫자를 total에 더한다
 - ③ '계속? Yes/no'를 묻는다
- total의 값을 출력한다

Solution : 파일명 yes_no_number_input.py



- ❖ 반복문에서 반복이 완료되기 전에 반복문을 빠져나오는 경우 사용
- ❖ 무한 루프 사용시 제어를 위해 break문을 사용
- ❖ break 문은 반복을 강제로 중단시키는 역할을 함

```
while True :  
    반복 문장  
    반복 문장  
    if 조건 :  
        break
```

<무한 루프 사용시 제어를 위해 break문을 사용>

```
a = 1  
while a <= 10:
```

```
.....  
.....  
if 조건 :  
    break  
.....  
.....  
a += 1
```

```
.....  
.....
```

```
a = 1  
while a <= 10:  
    print('a =', a)  
    if a >= 5 :  
        break  
    a += 1  
print('after while')
```

```
a = 1  
a = 2  
a = 3  
a = 4  
a = 5  
after while
```

- ❖ 5를 입력할 때까지 0~9 의 임의의 수 선택

```
import random

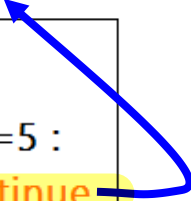
rand_num = 0

while True:
    rand_num = random.randint(0, 9)  # 0 ~ 9 사이에 임의의 수를 선택
    print(rand_num)
    if rand_num == 5:
        break
```

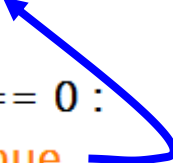
```
9
4
0
5
>>>
```

- ❖ **continue** 문은 반복문의 조건으로 제어가 가도록 한다.

```
a = 1
while a <= 10:
    .....
    .....
    if a >= 5 :
        continue
    .....
    .....
    a += 1
.....
.....
```



```
a = 0
while a <= 10:
    a += 1
    if a%3 == 0 :
        continue
    print('a =', a)
print('after while')
```



```
a = 1
a = 2
a = 4
a = 5
a = 7
a = 8
a = 10
a = 11
after while
```

continue 조건이 참인 경우 continue
아래 부분은 실행하지 않는다.

- ❖ 1부터 100사이의 숫자 중 홀수의 합이 1000이 넘지 않는 위치가 어디인지를 묻는 프로그램을 작성해보자.

```
sum = 0

for n in range(1,101,2):
    sum += n

    if sum > 1000 :
        break

print("1 + 3 + 5 + ... +", n-2, "=", sum-n)
```

```
sum = 0
n = 1
while True :
    sum += n
    if sum > 1000 :
        break
    n += 2
print("1 + 3 + 5 + ... +", n - 2, "=", sum - n)
```

- ❖ 프로그램이 선택한 정수를 사용자가 맞히는 게임
- ❖ 사용자가 답을 제시하면 프로그램의 정수와 비교하여 높은지 낮은지 만을 알려줌
- ❖ 게임이 끝나면 몇 번 만에 맞추었는지도 출력
- ❖ 시도횟수는 10번까지

1부터 100 사이의 숫자를 맞추시오
숫자를 입력하시오: 10
정답보다 낮음!
숫자를 입력하시오: 50
정답보다 높음!
숫자를 입력하시오: 40
정답보다 높음!
숫자를 입력하시오: 30
정답보다 높음!
숫자를 입력하시오: 20
정답보다 낮음!
숫자를 입력하시오: 25
축하합니다. 시도횟수= 6

Solution : 파일명 guess.py



Note: for문과 while문의 관계

- ❖ 반복문 for문과 while문은 상황에 따라 선택적으로 사용하지만 실질적으로는 동일한 반복문이기 때문에 언제든지 서로 간에 전환이 가능하다.
- ❖ 예: 1부터 100까지의 합을 구하는 프로그램

for문 사용시

```
sum = 0
for i in range(1, 101, 1):
    sum = sum + i
print(sum)
```

while문 사용시

```
sum = 0
i = 1

while i < 101:
    sum = sum + i
    i += 1

print(sum)
```

Note: for문과 while문의 관계

```
import turtle
geobuk = turtle.Turtle()
geobuk.shape('turtle')
```

```
for i in range(5):
    geobuk.forward(200)
    geobuk.left(144)
```



```
import turtle
geobuk = turtle.Turtle()
geobuk.shape('turtle')
```

```
i = 0
while i < 5:
    geobuk.forward(200)
    geobuk.left(144)
    i += 1
```