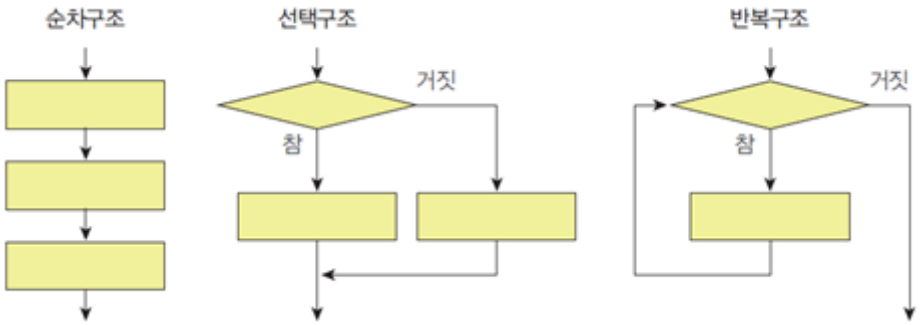
[3가지 기본 제어 구조]

- 순차 구조(sequence) - 명령들이 순차적으로 실행되는 구조 (연산자 ..)

- 선택 구조(selection) - 둘 중의 하나의 명령을 선택하여 실행 구조 (if 문)

- 반복 구조(iteration) - 동일한 명령이 반복되면서 실행되는 구조 (while, for 문)



반복문 필요성

In [ ]:
```
# 5개 출력
print("Please")
print("Please")
print("Please")
print("Please")
print("Please")
```

[반복문]

- 반복되는 일들을 처리하는 프로그램 필요

- 동일하거나, 규칙적으로 변화하는 작업

반복문 종류

1. (조건)에 따라 반복하는 while 문 : 반복조건

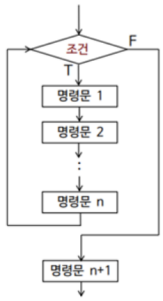
2. (지정된 범위)만큼 반복하는 for 문 : 반복횟수, 구간

- for 문 : 반복의 횟수가 미리 정해져 있는 경우

- while 문: 반복 횟수는 알지 못하지만 반복하는 조건이 명확한 경우에 사용

반복 논리

- 특정 명령 또는 연산을 반복적으로 수행해야 하는 경우
- 반복을 제어할 조건문이 있어야 한다.



◆ while 반복문

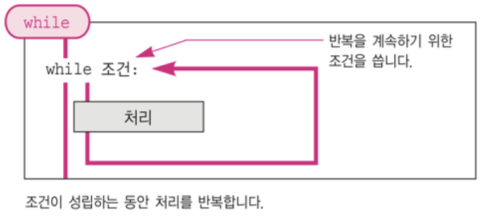
```
while 조건식 :
    명령어 1
    명령어 2
    …… 
    명령어 n
```

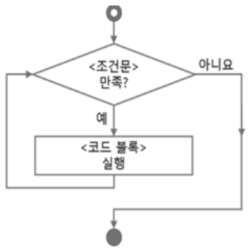
◆ for 반복문

```
for x in 데이터 :
    명령어 1
    명령어 2
    …… 
    명령어 n
```

[while문]

- 조건식 **True인 동안 내부구문 반복수행**
- 조건식은 반복 수행 이전에 우선 평가
- 구문을 모두 수행 이후 다시 조건식을 재평가
- 조건식이 False이면 while문 구조을 벗어남





In [ ]:
```
x = 0
while x < 5:
    print("Please")
    x += 1
```

In [ ]:
```
# while 문
x = 3
while x < 6:
    print(x)
    x = x + 1
```

In [ ]:
```
# while <조건문>: <수행할 문장1>
# else: <while 문이 종료된 후 수행할 문장1>
x = 3
while x < 6:
    print(x)
    x += 1
else:
    print("끝")
```

In [ ]:
```
# while True:
x = 3
while True:
    print(x)
    x += 1
    if x == 6: break
```

In [ ]:
```
# 1~100사이의 5의 배수
i = 1
```

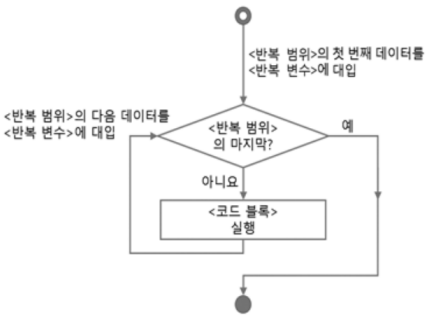
```
n = 1
while n < 100:
    n = i * 5
    print(n)
    i += 1
```

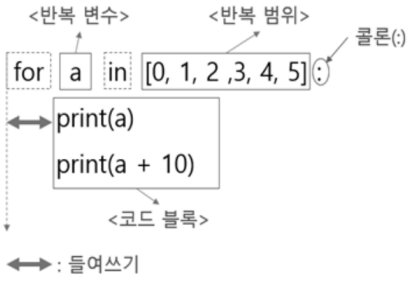
In [ ]:
```
## 구구단 계산
dan = int(input("원하는 단은 :"))
i = 1
while i <= 9:
    print("%d X % d = %2d" %(dan,i,dan*i))
    i += 1
```

In [ ]:
```
# 맞는 패스워드를 입력 할 때 까지 반복하기 (pwd:1234)
pwd = ""
while pwd != "1234" :
    pwd = input ("비밀번호를 입력하세요 : ")
```

[for 문 기본구조]

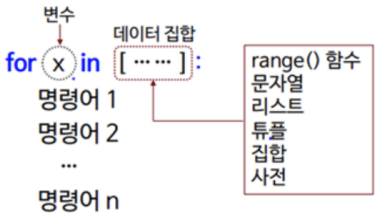
for ... in ... :





`for` item `in` iterable : 반복의 횟수, 범위내에서 실행

- iterable은 사전적의미와 똑같이 반복가능한 객체
- list, dictionary, set, string, tuple 가 iterable한 타입
- range, enumerate



- 데이터 집합에서 데이터를 하나씩 변수 x에 넣고 반복을 수행한다.
- for 구문에도 break와 continue를 사용할 수 있다.

In [ ]:
```
# range()
for looper in range(3):
    print("Hello")
```

In [ ]:
```
# range()
for i in range(4):
    print(i)
```

In [ ]:
```
# 문자열
for x in 'cat':
    print(x)
```

In [ ]:
```
# 튜플
for word in ('Alice', 'Bob', 'Carol'):
    print('Hello, %s' % word)
```

In [ ]:
```
# 리스트
for word in ['Alice', 'Bob', 'Carol']:
    print('Hello, %s' % word)
```

for 반복문 : 범위지정 방법

1. 컨테이너 자료형 - 반복가능한 (문자열, 리스트, 튜플, 집합, 사전) 사용 :

- for i in [0, 1, 2, 3, 4, 5]:

2. range() 함수 사용:

- `인수3개` : for i in range(시작,끝,스텝)
- `인수2개` : for i in range(시작,끝) : step 이 `1인 경우 생략 가능`
- `인수1개` : for i in range(끝) : step이 1이고 start가 0인 경우 start 역시 생략 가능

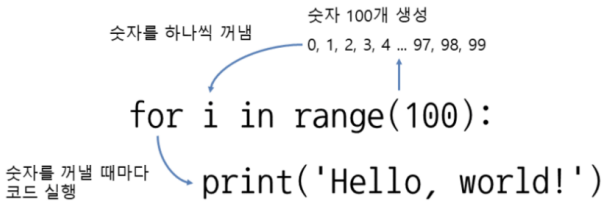
In [ ]:
```
# while()
i = 0
s = 0
while i <= 10:
    s = s + i
    i = i + 1
print(s)
```

In [ ]:
```
# for()
s = 0
for i in range(1, 11):
    s = s + i
print(s)
```

In [ ]:
```
# for()
s = 0
for i in [1,2,3,4,5,6,7,8,9,10] :
    s = s + i
print(s)
```

range( ) 활용 for 문

In [ ]:
```
for i in range(5):
    print('Hello, world!')
```



In [ ]:
```
a = list(range(5))
a
```

In [ ]:
```
for i in range(5):
    print(i)
```

In [ ]:
```
for i in list(range(5)):
    print(i)
```

In [ ]:
```
for i in [0, 1, 2, 3, 4]:
    print(i)
```

In [ ]:
```
# "*" 순차적으로 점점 많이 출력 (1-10개)
i = 0
for i in range(1, 11) :
    print('*' * i)
```

In [ ]:
```
# "*" 순차적으로 점점 적게 출력
i = 0
for i in range(10, 0, -1) :
    print('*' * i)
```

- 1에서 50 사이의 7의 배수와 그 수의 제곱 값을 출력하시오

In [ ]:
```
# 1에서 50 사이의 7의 배수와 그 수의 제곱 값을 출력하시오
for i in range(1, 50, 1) :
    if i%7 == 0 :
        print(i)
        print(i**2)
```

In [ ]:
```
# 1에서 50 사이의 7의 배수와 그 수의 제곱 값을 출력하시오
# if 사용하지 않고 가능
for i in range(7, 50, 7) :
    print(i)
    print(i**2)
```

In [ ]:
```
# range()와 for 반복문
for i in range(6):
    print(i, end=" ")
```

In [ ]:
```
# range()와 for 반복문
for i in range(2,6):
    print(i, end=" ")
```

In [ ]:
```
# range()와 for 반복문
for i in range(0,6,2):
```

```
    print(i, end=" ")
```

In [ ]:
```
# 덧셈, 단계별 출력
tot = 0
for i in range(1, 11):
    tot = tot + i
    print(tot, end ="  ")
```

In [ ]:
```
# for, if 혼합
for n in range(30):
    if n % 5 == 0:
        print (n)
```

In [ ]:
```
## 구구단 계산
dan = int(input("원하는 단은 :"))
for i in range(1,10,1):
    print("%d X % d = %2d" %(dan,i,dan*i))
```

list 활용 for문

for in 구문

- 반복문 키워드 for 와 in 사이에 계속 새롭게 할당할 변수 n을 선언
- in 뒤에 리스트 자료형을 넣어 리스트를 차례대로 순회하는 실행이 가능

In [ ]:
```
# 리스트와 for 반복문
numbers = [11, 22, 33, 44, 55, 66]
for n in numbers:
    print(n)
```

In [ ]:
```
# 리스트와 for 반복문 (친구 리스트 출력 )
myFriends = ['수빈', '용민', '진욱', '은석']  # 리스트를 변수에 할당
for f in myFriends:
    print(f)
```

In [ ]:
```
# 변수 x에 리스트 안에 원소 대입
animal = ["dog", "cat", "bird"]
for x in animal :
    print (x)
```

In [ ]:
```
# 리스트 안에 원소 갯수 만큼 반복
animal = ["dog", "cat", "bird"]
for x in animal :
    print ("반복")
```

In [ ]:
```
# 리스트와 for 반복문
animal = ["dog", "cat", "bird"]
for x in animal:
    print("I love " + x )
```

In [ ]:
```
# 문자열과 for 반복문
for i in 'hello':
    print(i)
```

In [ ]:
```
# 문자열과 for 반복문
for i in 'hello':
```

```
    print('abc')
    print(i)
```

In [ ]:
```
# 리스트와 for 반복문
for x in [1, 2, 3, 4, 5]:
#   print(x)
    print(x, end=" ")
```

In [ ]:
```
# 리스트와 for 반복문
for x in [3, 4, 5, 6]:
    print("hello",end=" " )
```

In [ ]:
```
# 튜플와 for 반복문
for x in (100, 200, 300) :
        print(x)
```

In [ ]:
```
# 리스트 항목내 정수 값들의 누적 덧셈
numbers = [10, 20, 30, 40, 50]
s = 0
for n in numbers:
    s = s + n
print('리스트 항목 값의 합 :', s)
```

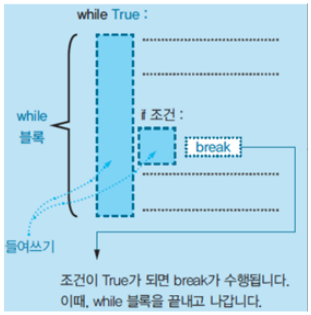
In [ ]:
```
# 성적 평균
a = [90, 85, 95, 80, 90, 100, 85, 75, 85, 80]
tot = 0
for i in range(len(a)): # len(a) : 리스트 a 원소 갯수
    tot = tot + a[i]
average = tot / len(a)
average
```

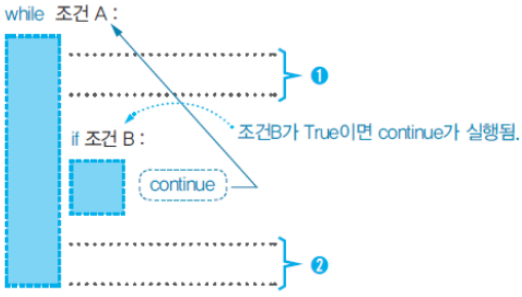
In [ ]:
```
# (리스트로 만들기)두 학생의 과목별 시험 성적의 합 구하기 (요소끼리 더하기)
tot = []
a1 = [90, 85, 95, 80, 90, 100, 85, 75, 85, 80]
a2 = [95, 90, 90, 90, 95, 100, 90, 80, 95, 90]

for i in range(len(a1)):
    tot.append(a1[i] + a2[i])
tot
```

[break 문, continune문]

- 무한 루프는 while True로 작성하고, break가 꼭 있어야 함

- continue 구문은 루프 안에서만 사용한다





- break문 : while문 강제로 빠져나가기

In [ ]:
```
# 맞는 패스워드를 입력 할 때 까지 반복하기 (pwd:1234)
pwd = ""
while True :
    pwd = input ("비밀번호를 입력하세요 : ")
    if pwd == "1234": break
```

In [ ]:
```
# [비교] 맞는 패스워드를 입력 할 때 까지 반복하기 (pwd:1234)
pwd = ""
while pwd != "1234" :
    pwd = input ("비밀번호를 입력하세요 : ")
```

In [ ]:
```
i = 0
while True:     # 무한 루프
    print(i)
    i += 1           # i를 1씩 증가시킴
    if i == 10:     # i가 10일 때
        break        # 반복문을 끝냄. while의 제어흐름을 벗어남
```

In [ ]:
```
for i in range(10000):     # 0부터 9999까지 반복
    print(i)
    if i == 10:     # i가 10일 때
        break        # 반복문을 끝냄. for의 제어흐름을 벗어남
```

- continue문 : while문 조건문으로 다시 으로 돌아가기

- while문을 빠져나가지 않고 while문의 맨 처음(조건문)으로 다시 돌아가게 만들고 싶은 경우

In [ ]:
```
# 홀수중 3을 제외
a = 0
while a < 10:
    a = a + 1
    if a%2 != 0:
        if a == 3: continue
        print(a)
```

In [ ]:
```
# 홀수중 3을 제외
for i in range(1,10,2) :
    if i == 3: continue
    print(i)
```

무한 루프

- **while문의 조건문이 True이므로 항상 참이 된다. 따라서 while문 안에 있는 문장들은 무한하게 수행될 것이다.**

```
while True:
    수행할 문장1
    수행할 문장2
    ...
```

In [ ]:
```
#while True:
#    print("Ctrl+C를 눌러야 while문을 빠져나갈 수 있습니다.")
```

In [ ]:
```python
i = 0
while True: # 조건문 참이면 반복수행 --> 무한반복
    print ('*' * i)
    i += 1  # while문 수행 시 1씩 증가
    if i > 5: break  # i 값이 4보다 크면 while문을 벗어난다.
```

---------- End ----------