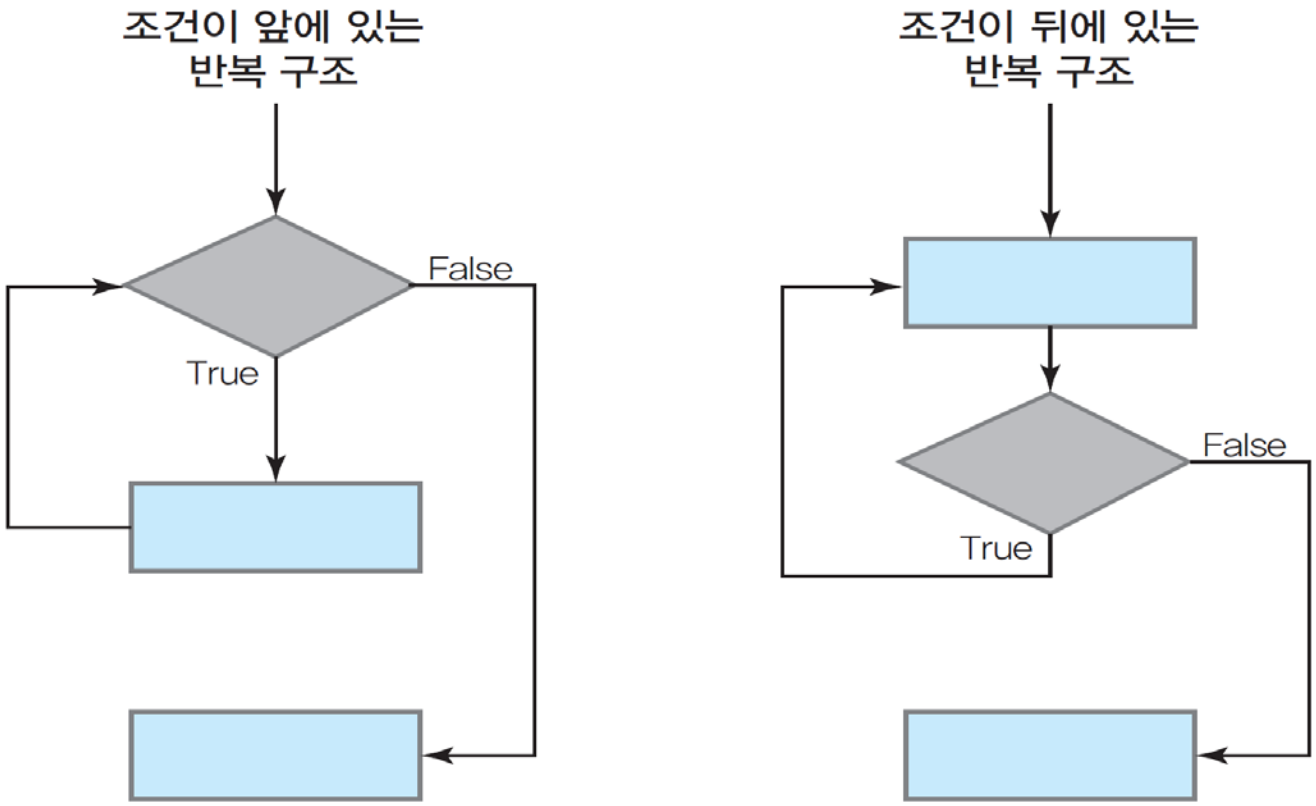
반복문
(Iteration Statement)

알고리즘의 구조: *반복 구조*

- 일련의 작업들을 여러 번 반복해야 하는 경우
  - 특정한 조건을 만족하는 동안 반복



반복(iteration) (1)

- 컴퓨터는 인간과 다르게 반복적인 작업을 실수 없이 빠르게 할 수 있다는 점이 큰 장점
  - *예) 회사에 중요한 손님이 오셔서 대형 전광판에 아래와 같이 "방문을 환영합니다!"를 반복해서 5번 출력*

```
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
```

복사해서 붙여넣기

```
방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!
```

반복(iteration) (2)

- *예) 만일 1000번 출력해야 된다면?*

```
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
print("방문을 환영합니다!")
        …
        …
print("방문을 환영합니다!")
```

1000번 복사해서 붙여넣기 ?

- 해결책: 반복 구조를 사용

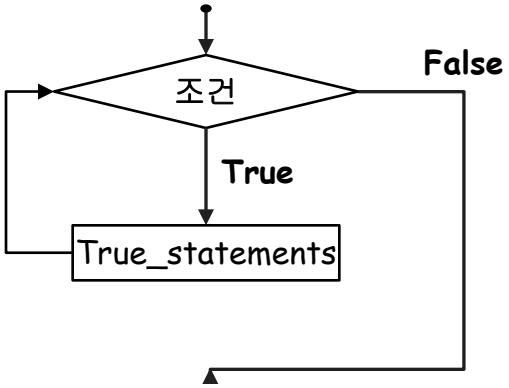
```
for i in range(1000):
    print("방문을 환영합니다!")
```

1000번 반복시키는 구조

반복문 유형: while문

- while문
  - *조건제어* 반복문
  - 특정 조건을 만족하는 동안 코드를 계속 반복해서 수행

while 조건 : 　　**True_statements**	조건을 만족하는 동안 True_statements을 반복적으로 수행

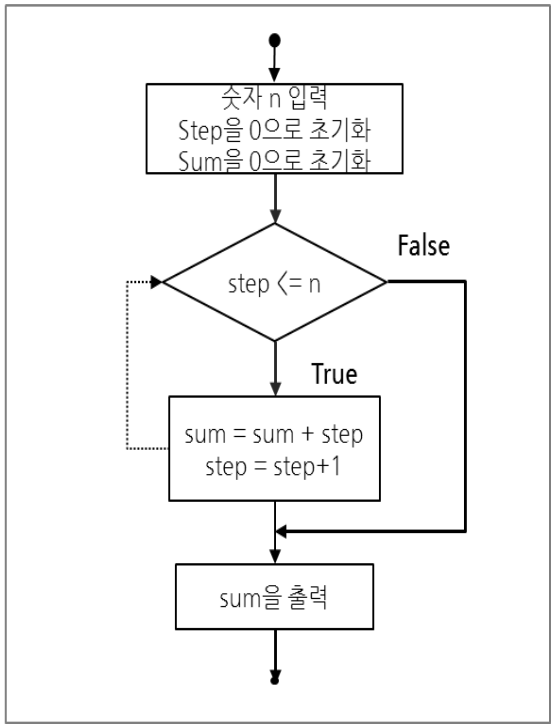


while문 예제

1부터 시작해 사용자가 입력한 값 까지 모든 정수를 더하는 프로그램을 만들어보자.

- 입력 값으로 20을 넣어서 1부터 20까지의 합을 출력해보자
- 입력 값으로 30을 넣어서 1부터 30까지의 합을 출력해보자

- 문제해결 알고리즘

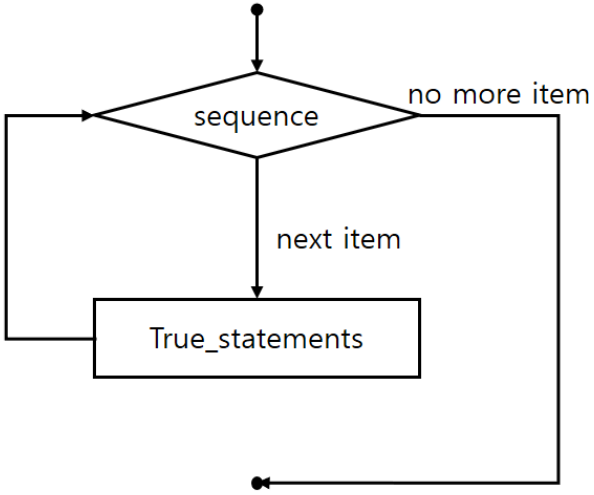



```
n = int(input('enter the number : '))

step = 1
sum = 0

while (step <= n) :
      sum += step
      step += 1

print(sum)
```

반복문 유형: for문

- for문
  - *횟수제어* 반복문
  - 반복의 횟수나 범위를 미리 알고 있을 경우에 사용하는 것이 효과적
  - 시작 값과 종료 값 등으로 범위를 지정하고 범위 내에서 반복을 수행

for item in sequence: ** True_statements**	**sequence** 내의 **item** 각각에 대해서 **True_statement**을 반복적으로 수행

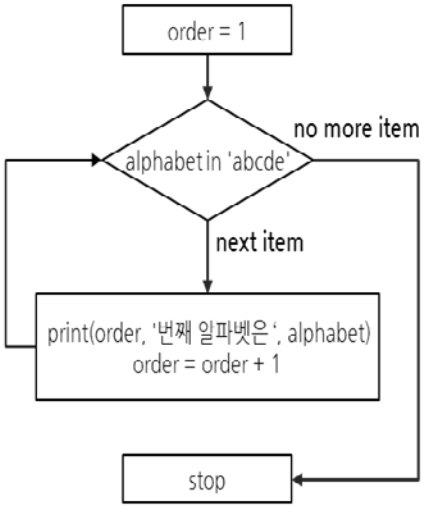


for문 예제

for character in 'string': True_statements	문자열 범위 내의 character 각각에 대해서 True_statements를 반복적으로 수행

1번째 알파벳 부터 5번째 알파벳 까지 알파벳 순으로 구성된
문사열 'abcde' 를 사용하여 다음과 같은 출력 결과를 나타내는
프로그램을 작성해보자

- 문제 해결 알고리즘




```
order = 1
for alphabet in 'abcde' :
      print(order,'번째 알파벳은', alphabet)
      order += 1
```

범위함수 range()를 활용한 for문

for variable in range(*num*): True_statements	range 범위 내의 variable 각각에 대해서 True_statements를 반복적으로 수행
range(*start, stop, step*)	*start*부터 *stop*전까지 *step*만큼씩 증가/감소하는 범위를 지정하는 함수

- *start*와 *stop*에는 정수만 쓸 수 있음
  - *step*은 0 일 수 없음
- *start* 값은 포함하지만 *stop*값은 포함되지 않음
  - 예) range (1, 9, 2)의 의미는 1, 3, 5, 7
- *start*와 *step*은 생략 될 수 있으나 *stop*은 생략될 수 없음
  - *start*가 생략되는 경우 *start* 값은 0이라고 간주
  - *step*이 생략되는 경우 *step* 값은 1이라고 간주
  - 예) range(5)의 의미는 0, 1, 2, 3, 4
  - 예) range(5, 10)의 의미는 5, 6, 7, 8, 9
  - 예) range (1, 10)의 의미는 1, 2, 3, …,9

range()를 활용한 for문 예제

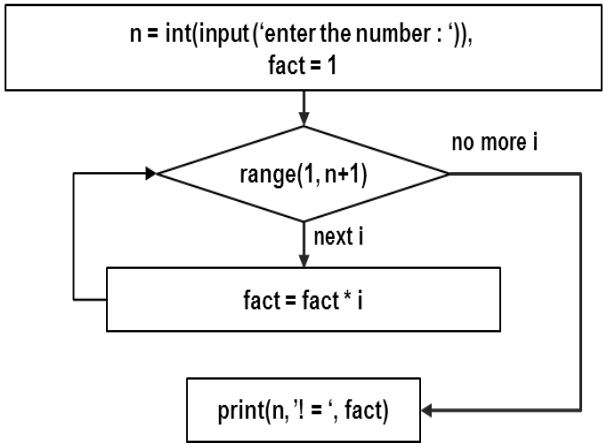
사용자가 **입력한 정수의 팩토리얼 값을 출력**하는 프로그램을 만들고자 한다.

입력 값을 **n으로 가정 했을 때**

n! = 1 × 2 × 3 × ... × n-1 × n 이다.

1) 5를 입력하고 결과를 출력해보자

- 문제 해결 알고리즘




```
n = int(input('enter the number : '))

fact = 1

for i in range(1,n+1) :
      fact =  fact * i

print(n, '! = ', fact)
```