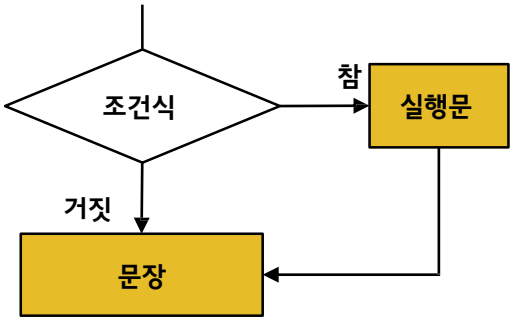
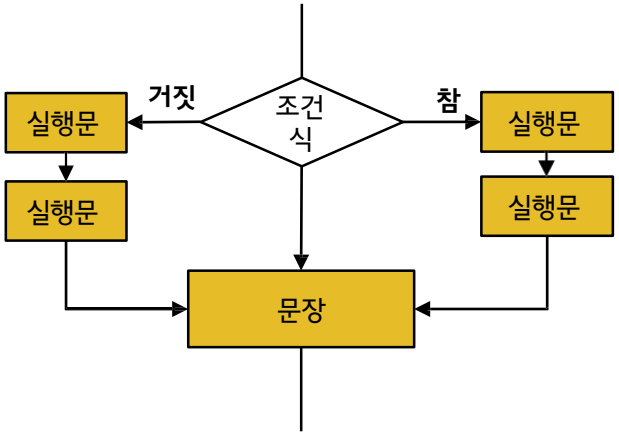
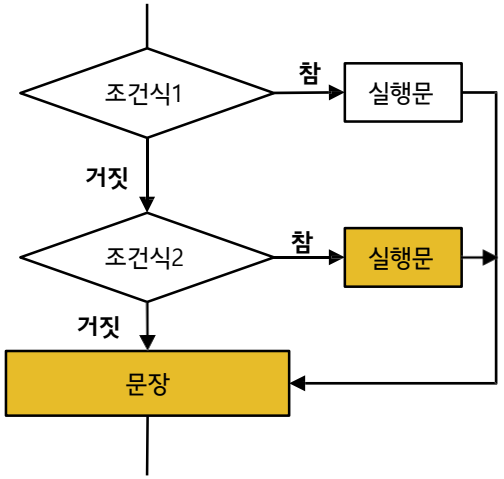
Python
제어문

제어-선택

- **조건문**
  - 프로그램의 실행 흐름을 바꾸고자 할 때 사용하는 제어문
  - 사용자가 지정한 조건에 따라 실행할 문장을 결정함







조건문 : if

- if 조건문 뒤에는 반드시 콜론(:)
  - 바로 아래 문장부터 if문에 속하는 모든 문장에 들여쓰기(indentation)를 해야 함
- 파이썬에는 다른 언어에 있는 switch 문이 존재하지 않음
  - if...elif...elif... 문으로 수행
- pass
  - def 문이나 if 문처럼 코드 블 록을 본문으로 갖는 표현에서 본문을 비워 둘 때 사용

```
#조건식
x = 10

print("조건식 1:")
if x % 2 == 1 :
    print(x, ": 홀수")

print("조건식 2:")
if x % 2 == 0 :
    print(x, ": 짝수")
else:
    print(x, ": 홀수")

print("조건식 3:")
if x <= 0 :
    print("양수가 아님")
elif x % 2 == 0 :
    print(x, ": 짝수")
else:
    print(x, ": 홀수")
```

```
조건식 1:
조건식 2:
10 : 짝수
조건식 3:
10 : 짝수
```

비교연산자/논리연산자

비교연산자	의미
x < y	x가 y보다 작다.
x > y	x가 y보다 크다.
x == y	x와 y가 같다.
x != y	x와 y가 같지 않다.
x >= y	x가 y보다 크거나 같다.
x <= y	x가 y보다 작거나 같다.

논리연산자	의미
and	x가 y보다 작다.
or	x가 y보다 크다.
not	x와 y가 같다.

```
#비교 연산자
x = 10
y = 20

print("비교 연산자")
print(x , '>', y , ":", x > y)
print(x , '<', y , ":", x < y)
print(x , '>=', y , ":", x >= y)
print(x , '>=', y , ":", x >= y)
print(x , '<=', y , ":", x <= y)
print(x , '==', y , ":", x == y)
print(x , '!=', y , ":", x != y)

print("논리 연산자")
if ( x >= 20 and y >= 20 ) : print ("두수는 20보다 크다")
elif ( x >= 20 or y >=20 ) : print("두수 중 20보다 큰수가 있다.")
else : print("두수는 20보다 작다")
```

```
비교 연산자
10 > 20 : False
10 < 20 : True
10 >= 20 : False
10 >= 20 : False
10 <= 20 : True
10 == 20 : False
10 != 20 : True
논리 연산자
두수 중 20보다 큰수가 있다.
```

in연산자

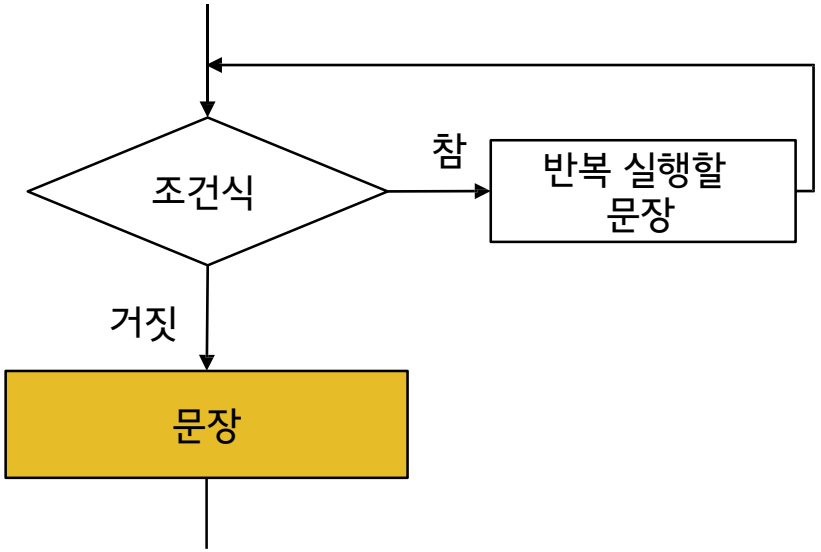
- **멤버십 연산자**
  - 문자열이나 리스트나 튜플과 같이 연속적인 자료 구조에 속한 멤버를 확인하기 위한 연산자

```
#in 연산자

str = "안녕하세요"
print(str, "에 안이란 글자가 있습니다.:", '안' in str)


#리스트
lst = list(range(1,5))
print("리스에 100이 있습니다.:", 10 in lst)
```

```
안녕하세요 에 안이란 글자가 있습니다.: True
리스에 100이 있습니다.: False
```

제어-반복

- **반복문**
  - 특정한 부분의 코드가 반복적으로 수행



반복문 - for

- **for**
  - 컬렉션으로부터 하나씩 요소(element)를 가져와, 루프 내의 문장들을 실행

```
#for문으로 collection 접근하기
#문자열
str = "안녕하세요"

print("문자열")
for item in str :
    print(item)

#리스트
lst = list(range(1,5))

print("리스트")
for item in lst :
    print(item)


#딕션너리
dic = {'a':1, 'b':2}

print("딕션너리")
for item in dic :
    print(item)

for item in dic.items():
    print(item)
```

```
문자열
안
녕
하
세
요
리스트
1
2
3
4
딕션너리
b
a
('b', 2)
('a', 1)
```

range() 함수와 반복문

- **range() 함수**
  - range(시작, 종료, step)
  - 결과는 시작숫자부터 종료 숫자 바로 앞 숫자까지 컬렉션 생성
  - 값을 확인하기 위해서는 순서가 있는 리스트나 튜플 컬렉션으로 변환해야 함
  - 시작숫자를 생략하면 0부터 생성
  - step을 생략하면 1씩 증가

```
xlist = range(10)
print(type(xlist))
print(xlist)

for item in xlist:
    print(item, end = ' ')

print()
print('-'*50)
xlist = list(range(10))
print(type(xlist))
print(xlist)

for item in xlist:
    print(item, end = ' ')
```

```
<class 'range'>
range(0, 10)
0 1 2 3 4 5 6 7 8 9
--------------------------------------
<class 'list'>
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
0 1 2 3 4 5 6 7 8 9
```

For문을 이용한 리스트 변형

```
st = ['1','2','3','4']
print("리스트 :", lst)

#for문을 이용한 리스트 항목의 형 변환
lst = [int(item) for item in lst]
print("리스트 :", lst)

#for문을 이용한 리스트 항목추출
lst = [item for item in lst if item % 2 == 0 ]
print("리스트 :", lst)
```

List Comprehension

[<결과식> for <변수명> in <순회할 수 있는 자료> if <조건식>]

```
>>> nums = [1,2,3,4]
>>> squares = [ x * x for x in nums ]
>>> squares
[1, 4, 9, 16]
>>> twice = [ 2 * x for x in nums if x > 2 ]
>>> twice
[6, 8]

>>> year = [2011, 2012, 2013, 2014, 2015]
>>> [i for i in year if i % 2 == 1]
[2011, 2013, 2015]
```

구구단 내포(comprehension)

```
# 이중 for문 이용

for i in range (2,10):
    for j in range(1,10):
        print(i*j, end= ‘ ‘)

# 리스트 내포 표현(list comprehension)

gu =[x*y for x in range(2,10) for y in range(1,10)]
print(gu)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 3, 6, 9, 12, 15, 18, 21,
24, 27, 4, 8, 12, 16, 20, 24, 28, 32, 36, 5, 10, 15, 20,
25, 30, 35, 40, 45, 6, 12, 18, 24, 30, 36, 42, 48, 54, 7,
14, 21, 28, 35, 42, 49, 56, 63, 8, 16, 24, 32, 40, 48, 56,
64, 72, 9, 18, 27, 36, 45, 54, 63, 72, 81]
```

짝수단만 출력

```
# 이중 for문 이용
for i in range (2,10):
    for j in range(1,10):
        if i % 2 == 0: # 짝수단일때
            print(i*j, end=‘ ’)
```

```
2 4 6 8 10 12 14 16 18 4 8 12 16 20 24 28 32 36 6 12 18 24
30 36 42 48 54 8 16 24 32 40 48 56 64 72
```

```
# 리스트 내포 표현(list comprehension)
gu=[i*j for i in range(2,10) for j in range(1,10) if i%2==0]
print(gu)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 4, 8, 12, 16, 20, 24, 28,
32, 36 6, 12, 18, 24, 30, 36, 42, 48, 54, 8, 16, 24, 32, 40,
48, 56, 64, 72]
```

반복문 while

- **while**
  - while 키워드 다음의 조건식이 참일 경우 계속 while 안의 블럭을 실행

```
#for문과 while문

print("for 문")
for i in range(1, 6) :
    print(i)

print("while 문")
i = 0
while i < 5 :
    i = i + 1
    print(i)
```

1~10까지 합

```
i, sum = 0, 0

while i <= 10:
    sum += i
    i += 1
    print(i, sum)
```

```
1 0
2 1
3 3
4 6
5 10
6 15
7 21
8 28
9 36
10 45
11 55
```

```
i, sum = 0, 0

while i <= 10:
    sum += i
    print(i, sum)
    i += 1
```

```
0 0
1 1
2 3
3 6
4 10
5 15
6 21
7 28
8 36
9 45
10 55
```

break/continue

- **break 문**
  - 반복문 안에서 루프를 빠져나오기 위해 사용

- **continue문**
  - 루프 블럭의 나머지 문장들을 실행하지 않고, 다음 루프로 직접 돌아가게 함

```
#break/continue
for i in range(1, 100) :
    if ( i % 2 == 1) : continue
    if ( i > 5 ) : break
    print(i)
```

```
2
4
```

해결문제

- 단을 입력 받아서 해당하는 단의 구구단을 출력하시오.
  - 단은 2단에서 9단까지만 입력 그 외 입력되면 종료하고 그렇지 않을 경우 계속 입력

```
단을 입력하세요6

****  6 단 ****
6  x  1  =  6
6  x  2  =  12
6  x  3  =  18
6  x  4  =  24
6  x  5  =  30
6  x  6  =  36
6  x  7  =  42
6  x  8  =  48
6  x  9  =  54


단을 입력하세요2

****  2 단 ****
2  x  1  =  2
2  x  2  =  4
2  x  3  =  6
2  x  4  =  8
2  x  5  =  10
2  x  6  =  12
2  x  7  =  14
2  x  8  =  16
2  x  9  =  18


단을 입력하세요99
입력오류
```