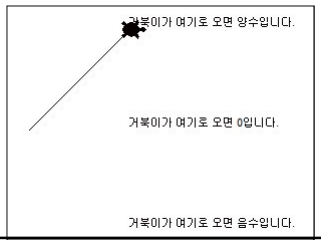
이번 장에서 만들 프로그램

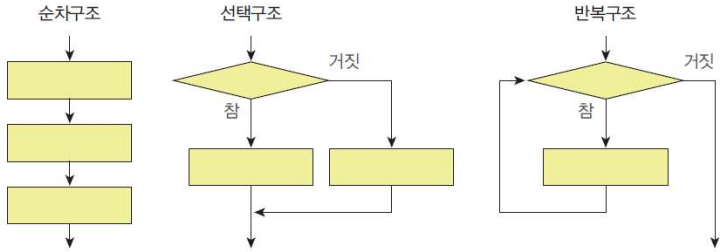
(1) 터틀 그래픽을 사용하여 동전의 앞면이나 뒷면이 나오는 동전 던지기 게임을 작성해보자.

(2) 정수의 부호에 따라서 거북이를 (100, 100), (100, 0), (100, -100)으로 움직이는 프로그램을 작성해보자.



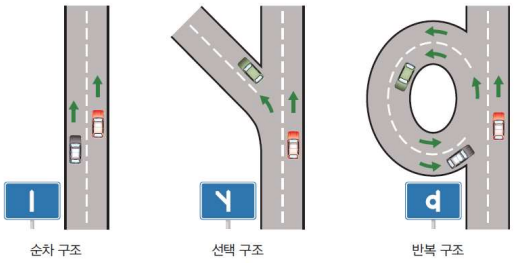
3가지의 기본 제어 구조

- 순차 구조(sequence) - 명령들이 순차적으로 실행되는 구조이다.
- 선택 구조(selection) - 둘 중의 하나의 명령을 선택하여 실행되는 구조이다.
- 반복 구조(iteration) - 동일한 명령이 반복되면서 실행되는 구조이다.



제어구조==도로

- 프로그램의 기본 블록을 쉽게 이해하려면 이것을 자동차(CPU)가 주행하는 도로로 생각하면 된다.



선택 구조가 필요한 이유

- 선택 구조가 없다면 프로그램은 항상 동일한 동작만을 되풀이할 것이다.
- (예) 자율 주행 자동차 프로그램이 신호등이나 전방 장애물에 따라서 동작을 다르게 하지 않는다면 어떻게 될까?

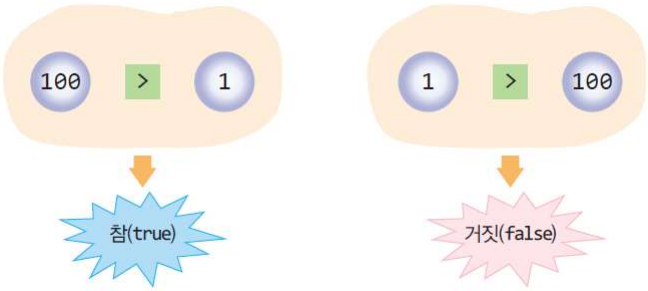
관계 연산자

- 관계 연산자(relational operator)는 두 개의 피연산자를 비교하는 연산자

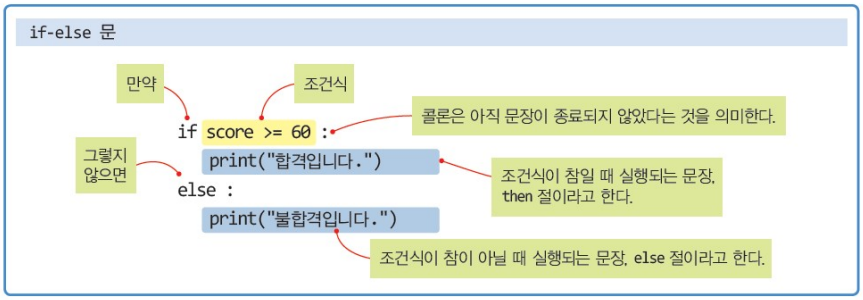
연산	의미
x == y	x와 y가 같은가?
x != y	x와 y가 다른가?
x > y	x가 y보다 큰가?
x < y	x가 y보다 작은가?
x >= y	x가 y보다 크거나 같은가?
x <= y	x가 y보다 작거나 같은가?

관계연산자의 결과값

- 관계 수식은 참(True)이나 거짓(False)을 생성한다.



if-else 문



예제 #1

```
score = int(input("성적을 입력하시오: "))
if score >= 60:
        print("합격입니다.")
else:
        print("불합격입니다.")
```

```
성적을 입력하시오: 80
합격입니다.
```

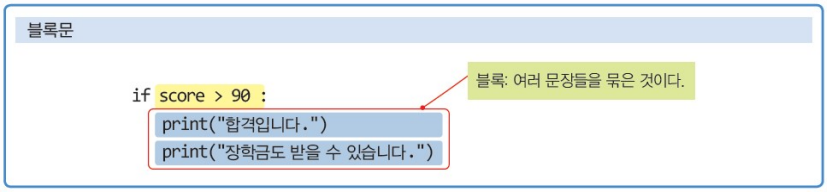
예제 #2

```
num = int(input("정수를 입력하시오: "))
if num % 2 == 0 :
        print("짝수입니다.")
else:
        print("홀수입니다.")
```

```
정수를 입력하시오: 10
짝수입니다.
```

블록

- 만약 조건이 참인 경우에 여러 개의 문장이 실행되어야 한다면 어떻게 해야 하는가?



Lab: 영화 나이 제한 검사

나이를 입력하시오: 19
이 영화를 보실 수 있습니다.

나이를 입력하시오: 14
이 영화를 보실 수 없습니다.

Solution

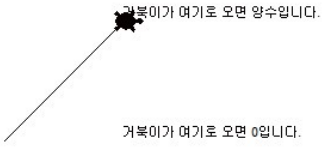
```
age = int(input("나이를 입력하시오: "))
if age >= 15:
        print("이 영화를 보실 수 있습니다.")
else:
        print("이 영화를 보실 수 없습니다.")
```

도전문제

15세 이상이면 "이 영화를 보실 수 있습니다." 메시지에 추가로 "영화의 가격은 10000원입니다."를 출력해보자. 만약 15세 미만이면 "이 영화를 보실 수 없습니다." 메시지에 추가로 "다른 영화를 보시겠어요?"를 출력해보자.

Lab: 부호에 따라 거북이를 움직이자

- 사용자로부터 정수를 받아서 정수의 부호에 따라서 거북이를 (100, 100), (100, 0), (100,-100)으로 움직이는 프로그램을 작성해보자.



```
import turtle
t = turtle.Turtle()
t.shape("turtle")

t.penup() # 펜을 올려서 그림이 그려지지 않게 한다.
t.goto(100, 100) # 거북이를 (100, 100)으로 이동시킨다.
t.write("거북이가 여기로 오면 양수입니다.")
t.goto(100, 0)
t.write("거북이가 여기로 오면 0입니다.")
t.goto(100, -100)
t.write("거북이가 여기로 오면 음수입니다.")

t.goto(0, 0) # (0, 0) 위치로 거북이를 이동시킨다.
t.pendown() # 펜을 내려서 그림이 그려지게 한다.
s = turtle.textinput("", "숫자를 입력하시오: ")
n=int(s)
if( n > 0 ):
        t.goto(100, 100)
if( n == 0 ):
        t.goto(100, 0)
if( n < 0 ):
        t.goto(100, -100)
```

논리 연산자

조건 1 조건 2

나이가 10살 이상이고, 그리고 키가 165cm 이상이면
→ 놀이기구를 탈 수 있다.

(나이가 10살 이상이다) and (키가 165cm 이상이다)
→ 놀이기구를 탈 수 있다.

그리고

age >= 10 and height >= 165

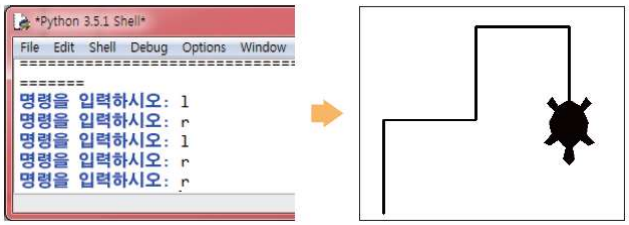
나이가 10살 이상 키가 165cm 이상

논리 연산자의 종류

연산	의미
x and y	AND 연산, x와 y가 모두 참이면 참, 그렇지 않으면 거짓
x or y	OR 연산, x나 y중에서 하나만 참이면 참, 모두 거짓이면 거짓
not x	NOT 연산, x가 참이면 거짓, x가 거짓이면 참

Lab: 거북이 제어하기

- 파이썬 쉘에서 "l"을 입력하면 거북이가 왼쪽으로 100픽셀 이동하고 "r"을 입력하면 거북이가 오른쪽으로 100픽셀 이동하는 프로그램을 작성해



Run Python

무한 반복 구조

- 아직 학습하지 않았지만 다음과 같은 코드를 사용하면 무한 반복할 수 있다.

```
while True:
        ...
        ...
        ...
```

Solution

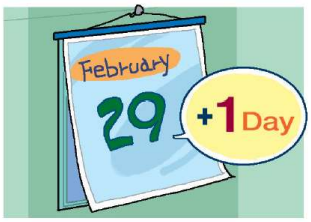
```
import turtle

# 거북이를 만든다.
t = turtle.Turtle()
# 거북이가 그리는 선의 두께를 3으로 한다.
t.width(3)
# 커서의 모양을 거북이로 한다.
t.shape("turtle")
# 거북이를 3배 확대한다.
t.shapesize(3, 3)

# 무한 루프이다.
while True:
        command = input("명령을 입력하시오: ")
        if command == "l": # 사용자가 "l"을 입력하였으면
                t.left(90)
                t.forward(100)
        if command == "r": # 사용자가 "r"을 입력하였으면
                t.right(90)
                t.forward(100)
```

Lab: 윤년 판단

- 입력된 연도가 윤년인지 아닌지를 판단하는 프로그램을 만들어 보자.

```
연도를 입력하시오: 2012
2012 년은 윤년입니다.
```

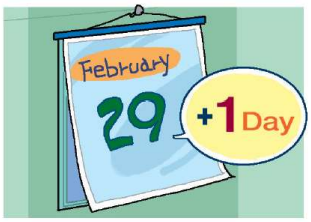
✓ 연도가 4로 나누어 떨어지면 윤년이다.
✓ 100으로 나누어 떨어지는 연도는 제외한다.
✓ 400으로 나누어 떨어지는 연도는 윤년이다.



윤년의 조건

```
( (year % 4 == 0) and ( year % 100 != 0 ) ) or ( year % 400 == 0 )
```

- `(year % 4 == 0)`: 연도가 4로 나누어떨어진다.
- `year % 100 != 0`: 100으로 나누어떨어지는 연도는 제외한다.
- `year % 400 == 0`: 400으로 나누어떨어지는 연도는 윤년이다.



Solution

```
year = int(input("연도를 입력하시오: "))
if ( (year % 4 ==0 and year % 100 != 0) or year % 400 == 0):
        print(year, "년은 윤년입니다.")
else :
        print(year, "년은 윤년이 아닙니다.")
```

FEBRUARY 2012						
SUN	MON	TUE	WED	THU	FRI	SAT
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29			

Lab: 동전 던지기 게임

- 동전을 던지기 게임을 작성해 보자.
- import random한 후에 random.randrange(2) 하면 0이나 1을 랜덤하게 생성할 수 있다.

동전 던지기 게임을 시작합니다.
뒷면입니다.
게임이 종료되었습니다.

Run Python

Solution

```
import random

print("동전 던지기 게임을 시작합니다.")
coin = random.randrange(2)
if coin == 0 :
        print("앞면입니다.")
else :
        print("뒷면입니다.")
print("게임이 종료되었습니다.")
```

Lab: 동전 던지기 게임(그래픽 버전)

- 동전을 던지기 게임을 그래픽 버전으로 만들어보자.

또는

이미지를 불러오려면

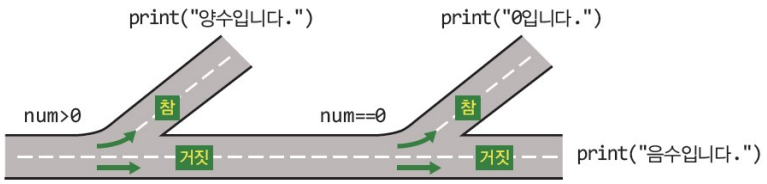
```
screen = turtle.Screen()
image1 = "d:\\front.gif"
image2 = "d:\\back.gif"
screen.addshape(image1) # 이미지를 추가한다.
screen.addshape(image2) # 이미지를 추가한다.
t1.shape(image1) # 거북이의 모양을 설정한다.
t1.stamp() # 현재 위치에 거북이를 찍는다.
```

Solution

```
import turtle # 터틀 그래픽 모듈을 불러온다.
import random # 난수 모듈을 불러온다.

screen = turtle.Screen()
image1 = "d:\\front.gif"
image2 = "d:\\back.gif"
screen.addshape(image1)
screen.addshape(image2)

t1 = turtle.Turtle() # 첫 번째 거북이를 생성한다.
coin = random.randint(0, 1)
if coin == 0 :
        t1.shape(image1)
        t1.stamp()
else :
        t1.shape(image2)
        t1.stamp()
```

조건을 연속하여 검사

- 다음과 같이 진행하는 코드를 작성하려면?



연속적인 if-else 문

```
num = int(input("정수를 입력하시오: "))

if num > 0:
        print("양수입니다.")
elif num == 0:
        print("0입니다.")
else:
        print("음수입니다.")
```

```
정수를 입력하시오: 10
양수입니다.
```

Lab: 종달새가 노래할까?

- 동물원에 있는 종달새가 다음과 같은 2가지 조건이 충족될 때 노래를 한다고 하자.
  - 오전 6시부터 오전 9시 사이
  - 날씨가 화창하다.

난수 이용

- 현재 시각을 난수로 생성하고 날씨도 [True, False] 중에서 랜덤하게 선택하자. 종달새가 노래를 부를 것인지, 조용히 있을 것인지를 판단해보자.

```
import random
time = random.randint(1, 24)
sunny = random.choice([True, False])
```

```
좋은 아침입니다. 지금 시각은 1시 입니다.
현재 날씨가 화창하지 않습니다.
종달새가 노래를 하지 않는다.
```

Solution

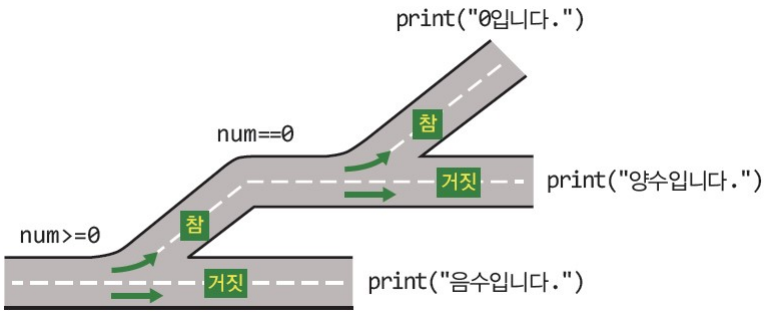
```
import random
time = random.randint(1, 24)
print("좋은 아침입니다. 지금 시각은 " + str(time) + "시 입니다.")

sunny = random.choice([True, False])
if sunny:
        print ("현재 날씨가 화창합니다. ")
else:
        print ("현재 날씨가 화창하지 않습니다. ")

# 종달새가 노래를 할 것인지를 판단해보자.
if time >= 6 and time < 9 and sunny:
        print ("종달새가 노래를 한다.")
else
        print ("종달새가 노래를 하지 않는다.")
```

도전문제

종달새가 6시와 9시 사이 또는 14시와 16시 사이에 노래한다고 하면(그리고 날씨는 항상 화창하여야 한다) 위 조건식을 어떻게 변경하여야 하는가?

중첩 if-else문

- if 문 안에 다른 if 문이 들어갈 수도 있다. 이것을 중첩 if 문이라고 한다.



예제

```
num = int(input("정수를 입력하시오: "))
if num >= 0:
        if num == 0:
                print("0입니다.")
        else:
                print("양수입니다.")
else:
        print("음수입니다.")
```

```
정수를 입력하시오: 10
양수입니다.
```

Lab: 로그인 프로그램

- 사용자로부터 아이디를 받아서 프로그램에 저장된 아이디와 일치하는지 여부를 출력하는 프로그램을 작성해보자.

```
아이디를 입력하시오: ilovepython
환영합니다.
```

```
아이디를 입력하시오: iloveruby
아이디를 찾을 수 없습니다.
```

Solution

```
id = "ilovepython"
s = input("아이디를 입력하시오: ")
if s == id:
        print("환영합니다.")
else:
        print("아이디를 찾을 수 없습니다.")
```

도전문제

아이디 검사가 종료되면 바로 패스워드 검사를 해보자. 즉 다음과 같은 출력을 가지는 프로그램을 작성한다.

```
아이디를 입력하시오: ilovepython
패스워드를 입력하시오: 123456
환영합니다.
```

Lab: 축구게임

- 사용자로부터 아이디를 받아서 프로그램에 저장된 아이디와 일치하는지 여부를 출력하는 프로그램을 작성해 보자.

```
어디를 수비하시겠어요?(왼쪽, 중앙, 오른쪽)중앙
페널티 킥이 성공하였습니다.
```

Run Python

Solution

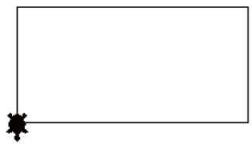
```
options=["왼쪽","중앙","오른쪽"]
computer_choice = random.choice(options)
user_choice = input("어디를 수비하시겠어요?(왼쪽, 중앙, 오른쪽)")
if computer_choice == user_choice:
        print("수비에 성공하셨습니다. ")
else:
        print("페널티 킥이 성공하였습니다. ")
```

도전문제

골대를 더 여러 영역으로 나누어서 게임을 다시 작성해보자. 즉 왼쪽 상단, 왼쪽 하단, 중앙, 오른쪽 상단, 오른쪽 하단 중에서 하나를 선택하도록 하라.

Lab: 도형그리기

- 터틀 그래픽을 이용하여 사용자가 선택하는 도형을 화면에 그리는 프로그램을 작성해보자. 도형은 "사각형", "삼각형", "원" 중의 하나이다. 각 도형의 치수는 사용자에게 물어보도록 하자.



Run Python

Solution

```
import turtle
t = turtle.Turtle()
t.shape("turtle")

s = turtle.textinput("", "도형을 입력하시오: ")
if s == "사각형" :
        s = turtle.textinput("","가로: ")
        w=int(s)
        s = turtle.textinput("","세로: ")
        h=int(s)
        t.forward(w)
        t.left(90)
        t.forward(h)
        t.left(90)
        t.forward(w)
        t.left(90)
        t.forward(h)
```

도전문제

위의 프로그램에서 "사각형"만을 지원하고 있다. "삼각형", "원"인 경우에 도형을 그리는 코드를 추가하라.

이번 장에서 배운 것

- *>, <, ==와 같은 관계 연산자를 학습하였다.*
- *논리 연산자 and나 or 를 사용하면 조건들을 묶을 수 있다.*
- *블록은 조건이 맞았을 때 묶어서 실행되는 코드로 파이썬에서 들여쓰기로 블록을 만든다.*
- *if-else 문 안에 다른 if-else 문이 포함될 수 있다.*

Q & A