파이썬 Express

3장 조건문

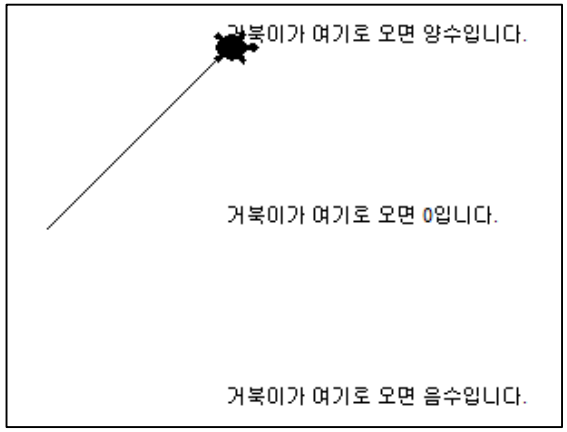
학습 목표

- 제어문에 대하여 이해합니다.
- if-else 문을 이해하고 사용할 수 있습니다.
- 관계연산자와 논리연산자를 학습합니다.
- 블록의 개념을 학습합니다.
- 중첩 if-else 문을 학습합니다.
- 연속 if-else 문을 학습합니다.

이번 장에서 만들 프로그램

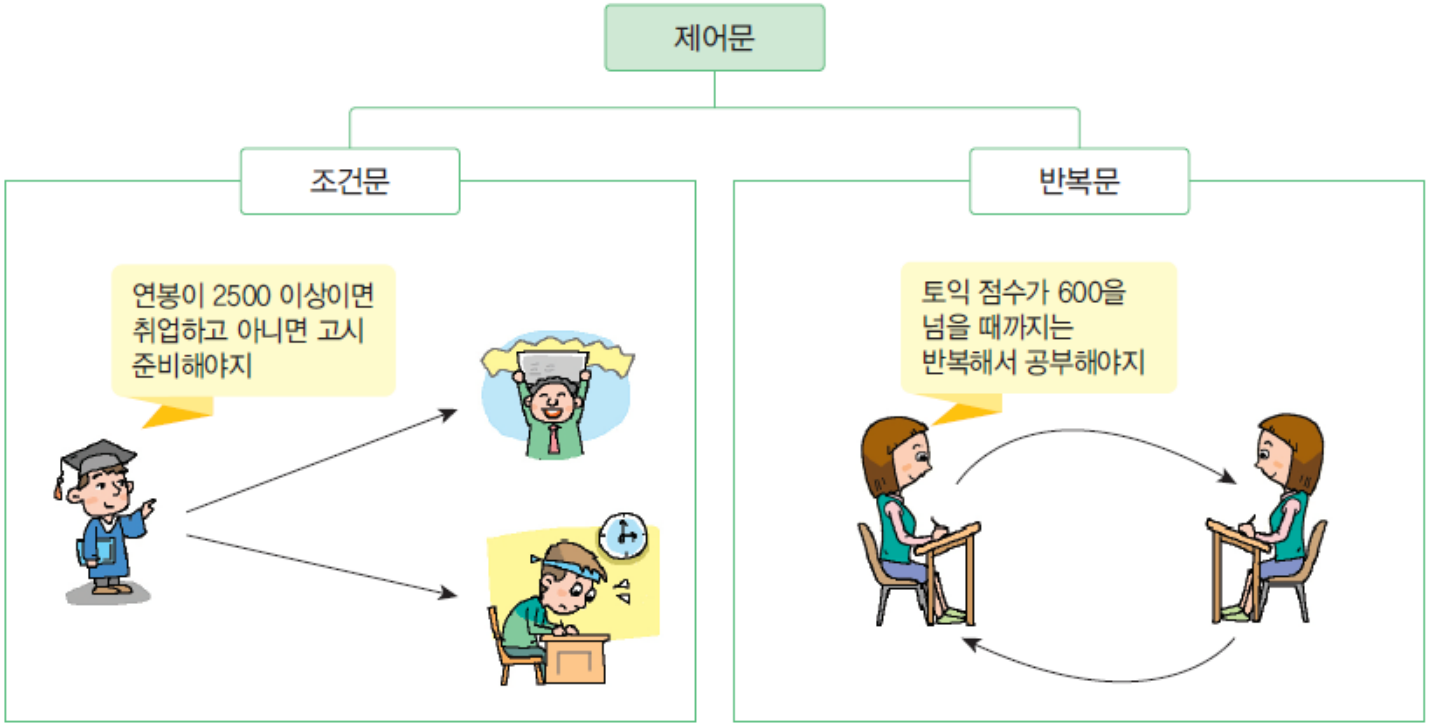
```
정가를 입력하시오: 200
10층에서 사은품을 받아가세요.
15% 할인된 가격= 170.0
```

```
리히터 규모를 입력하시오: 5.2
빈약한 건물에 큰 피해가 있습니다.
```



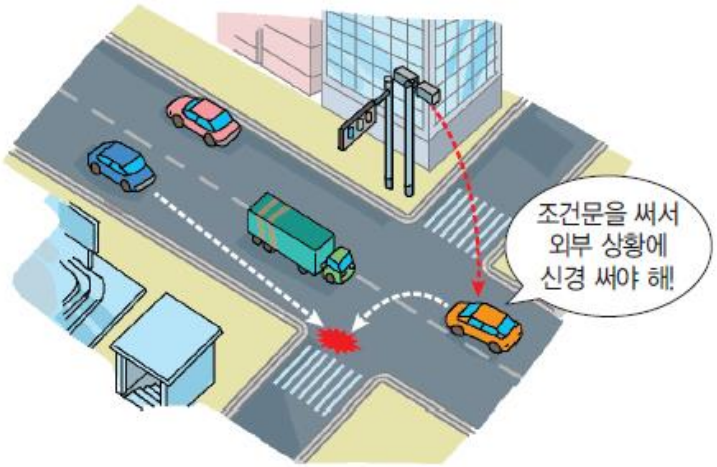
제어문

- 프로그램으로 현실세계의 복잡한 문제를 처리하려면 조건에 따라서 실행을 다르게 하거나 조건에 따라서 반복할 수 있어야 한다. 문장들이 실행되는 순서를 제어하는 문장을 제어문(control statement)라고 한다.

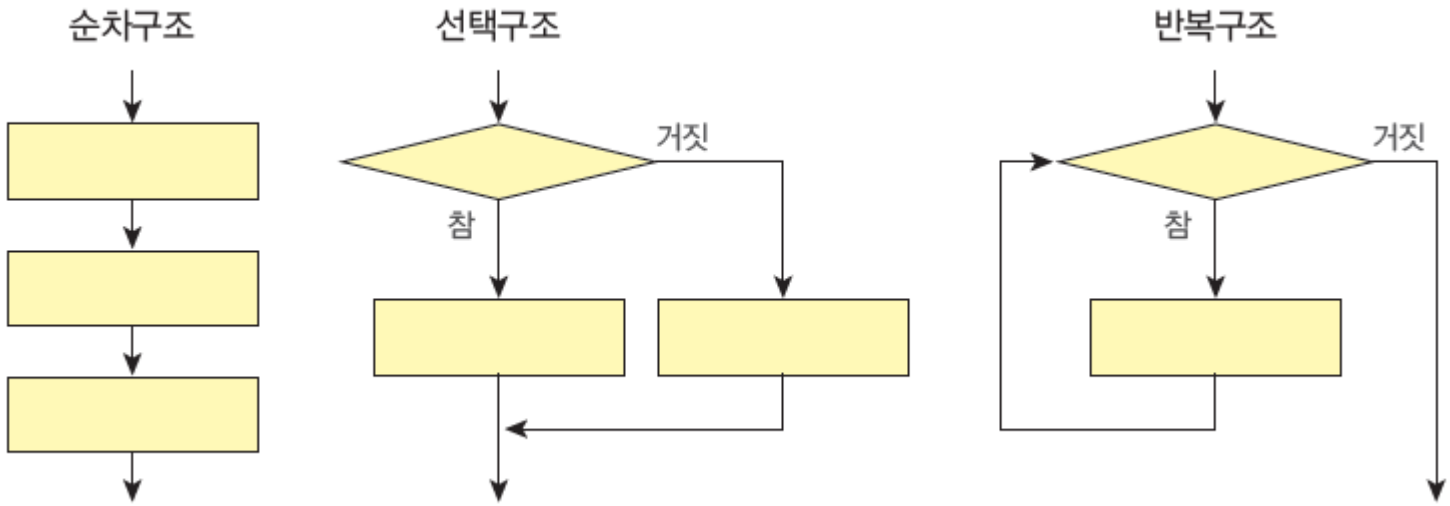


조건문

- 어떤 조건에 따라서 문장의 실행 여부가 결정되는 명령문
- 만약 프로그램에 조건문이 없다면 프로그램은 항상 동일한 동작만을 되풀이 할 것이다. 자율 주행 자동차가 센서의 신호에 따라 동작을 다르게 하지 않는다면 큰일이 날 것이다.



3가지의 제어구조

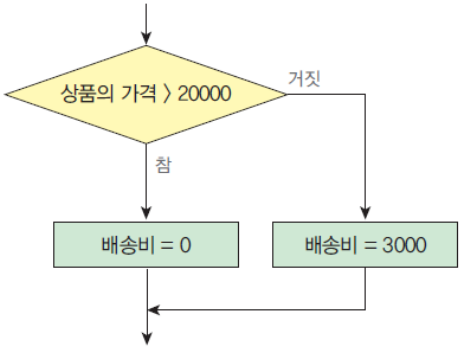


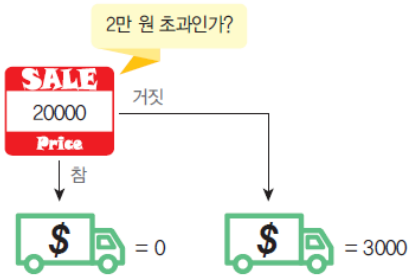
중간점검

1. 프로그램에 사용되는 3가지의 제어구조는 어떤 것들인가?
2. 제어문은 _____과 _____으로 나누어진다.

if-else 문

- 쇼핑몰에서 상품의 가격이 2만원을 넘으면 배송비는 없고 그렇지 않으면 3000원의 배송비가 붙는 경우.





Syntax: if-else 문

형식
```
if 조건식 :
        문장1
else :
        문장2
```

예
```
if price > 20000 :
        shipping_cost = 0
else :
        shipping_cost = 3000
```

- 참이나 거짓으로 계산되는 조건식. 관계 연산자 == != 〈 〉 >= <=을 사용한다.
- 콜론(:)은 복합문을 의미한다.
- 조건식이 참이면 실행되는 문장
- 조건식이 거짓이면 실행되는 문장
- else절은 생략될 수도 있다.
- if와 else는 같은 위치여야 한다.

배송비 계산 프로그램

p118_shipping.py

```
# 사용자로부터 상품의 가격을 입력받는다.
price = int(input("상품의 가격: "))

# 배송비를 결정한다.
if  price > 20000 :
        shipping_cost = 0
else :
        shipping_cost = 3000

# 배송비를 출력한다.
print("배송비 = ", shipping_cost)
```

```
상품의 가격: 30000
배송비 =  0
```

블록

- 만약 조건이 참인 경우에 여러 개의 문장이 실행되어야 한다면 -> 들여쓰기를 이용하여서 문장들을 묶을 수 있다 -> 블록(block)

```
if price > 20000 :
    shipping_cost = 0
    discount = 0.1
else :
    shipping_cost = 3000
```

블록: 여러 문장들을 묶은 것이다.

```
if price > 20000 :
□□□□shipping_cost = 0
□□□□discount = 0.1
else :
    shipping_cost = 3000
```

블록

else는 없을 수도 있다.

- If-else 문을 사용할 때 만약 else 부분이 필요없다면 생략할 수 있다.

```
shipping_cost = 3000                # 기본적으로 배송비는 3000원이다.
if  price > 20000 :                 # 만약 상품의 가격이 2만원 초과이면
        shipping_cost = 0           # 배송비가 없다.
```

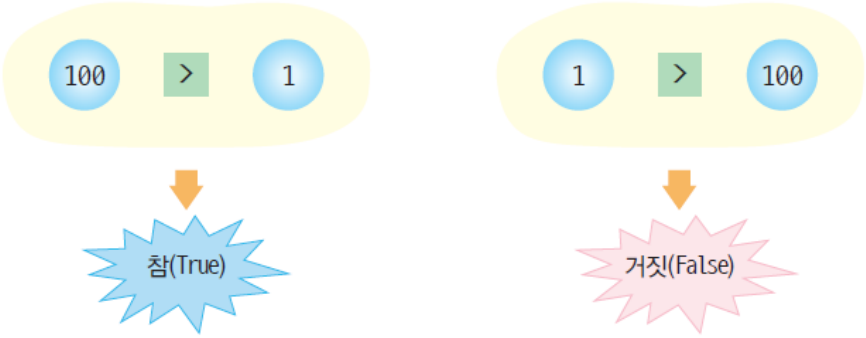
파이썬에서는 들여쓰기가 아주 중요하다. if-else 문에서도 들여쓰기가 잘못되면 오류가 발생한다.

```
if number > 0 :
        print("양수")
else :
        print("음수")

0       1    들여쓰기 레벨
```

관계 연산자(relational opterator)

- If-else 문에서 조건을 나타내기 위한 연산자가 관계 연산자이다.

연산	의미	수학적 표기
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?	≤



부울 변수(Boolean value)

- 관계 연산자의 결과값은 True 아니면 False인 부울값이다.

```
radius = 100
flag = (radius > 32)
print(flag)
```

```
True
```

p122_bool.py

```
expensive = price > 20000          # expensive가 부울 변수이다.
if  expensive :                    # 관계 수식 대신에 부울 변수가 들어가도 된다.
        shipping_cost = 0
else :
        shipping_cost = 3000
```

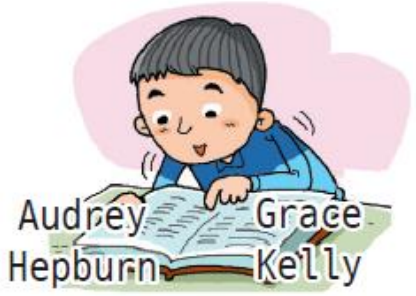
문자열 비교

```
s1 = "Audrey Hepburn"
s2 = "Audrey Hepburn"
print(s1 == s2)
```

```
True
```

```
s1 = "Audrey Hepburn"
s2 = "Grace Kelly"
print(s1 < s2)
```

```
True
```



- 기본적으로 사전 순서이지만 주의할 점도 있다.
- 모든 대문자는 소문자 앞에. 스페이스 문자는 모든 알파벳 문자보다 앞에. 숫자는 문자보다 앞에.

실수와 실수의 비교

p123.py

```
from math import sqrt

n = sqrt(3.0)
if n*n == 3.0 :
        print("sqrt(3.0)*sqrt(3.0)은 3.0과 같다. ")
else :
        print("sqrt(3.0)*sqrt(3.0)은 3.0과 같지 않다. ")
```

```
sqrt(3.0)*sqrt(3.0)은 3.0과 같지 않다.
```

- 실수와 실수를 비교할 때는 아주 조심해야 한다. 컴퓨터는 비트의 수가 제한되어 있어 복잡한 값은 정확히 표현되지 않기 때문. 실수 2개가 같은지를 판별하려면 다음과 같이 차를 감안하여서 비교하여야 한다.

```
if abs(n*n - 3.0) < 0.00001 :
        print("sqrt(3.0)*sqrt(3.0)은 3.0과 같다. ")
```

```
sqrt(3.0)*sqrt(3.0)은 3.0과 같다.
```

Lab: 산술 퀴즈 프로그램

- 산수 퀴즈를 발생시키는 프로그램

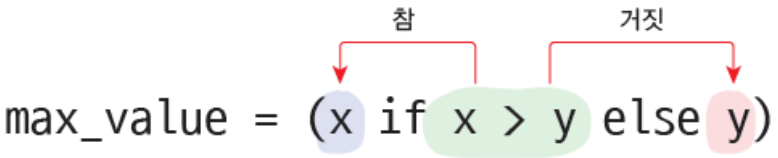
```
25 + 78 = 103
True
```

```
25 + 78 = 100
False
```

p124_mathprob.py

```
##
#          이 프로그램은 산수 문제를 출제한다.
#

import random

x = random.randint(1, 100)
y = random.randint(1, 100)

answer = int(input(f"{x} + {y} = "))

# 부울 변수에 결과를 저장하고 출력한다.
flag = (answer == (x+y))
print(flag)
```

조건 연산자




```
예.
shipping_cost = ( 0 if price >= 20000 else 3000 )

absolute_value = (x if x > 0 else -x)        // 절대값 계산
max_value = (x if x > y else y)              // 최대값 계산
min_value = (x if x < y else y)              // 최소값 계산
```

p126_conditional.py

```
x = int(input("첫 번째 수 ="))
y = int(input("두 번째 수 ="))
max_value = (x if x > y else y)
min_value = (y if x > y else x)
print("큰 수=", max_value, "작은 수=", min_value)
```

```
첫 번째 수 =10
두 번째 수 =20
큰 수= 20 작은 수= 10
```

Lab: 짝수와 홀수를 구분해보자

```
정수를 입력하시오: 10
짝수입니다.
```

p126_even_odd.py

```
number = int(input("정수를 입력하시오: "))

if number % 2 == 0 :
    print("짝수입니다.")
else:
    print("홀수입니다.")
```

도전문제

1. 사용자로부터 받은 정수가 양수인지 음수인지를 구별하는 프로그램을 작성하라. 0은 양수로 간주한다.

2. 프로그램에서 사용자의 성적을 입력받는다. 만약 입력된 값이 60 이상이면 "합격입니다."를 출력하고, 그렇지 않으면 "불합격입니다." 메시지를 출력하는 프로그램을 작성하라.

Lab: 세일 가격 계산

- 상품의 가격이 100만원 미만이면 10% 할인이 적용된다. 만약 상품의 가격이 100만원 이상이면 15%의 할인이 적용된다. 그리고 100만원 이상의 상품을 사면 사은품이 지급된다. 100만원 미만이면 사은품은 없다.

```
정가를 입력하시오: 200
10층에서 사은품을 받아가세요.
할인된 가격= 170.0
```

```
정가를 입력하시오: 80
할인된 가격= 72.0
```

p129_sale.py

```
price = int(input("정가를 입력하시오: "))
if price >= 100 :
        dis_rate = 0.85
        print("10층에서 사은품을 받아가세요.")
else :
        dis_rate = 0.90
dis_price = dis_rate * price
print("할인된 상품의 가격=", dis_price)
```

논리 연산자

- 복잡한 조건을 표현할 때

상품의 가격이 2만원 초과, 그리고 "파이썬" 카드이면
-> 배송료가 없음

(상품의 가격이 2만원 초과이다) and ("파이썬" 카드이면)
-> 배송료가 없음

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

예제

p131_logic.py

```
price = int(input("가격을 입력하시오: "))
card = input("카드 종류를 입력하시오: ")

if price > 20000 and card == "python" :
        print("배송료가 없습니다.")
else
        print("배송료는 3000원입니다.")
```

```
가격을 입력하시오: 30000
카드 종류를 입력하시오: python
배송료가 없습니다.
```

```
가격을 입력하시오: 30000
카드 종류를 입력하시오: java
배송료는 3000원입니다.
```

드모르간의 법칙

- 인간은 일반적으로 **not** 연산자가 적용된 수식을 이해하기가 어렵다. 논리 학자 드모르간(De Morgan)의 이름을 딴 드모르간의 법칙을 사용하여 이러한 논리식을 단순화할 수 있다.

$$\neg(P \vee Q) \iff (\neg P) \wedge (\neg Q),$$

$$\neg(P \wedge Q) \iff (\neg P) \vee (\neg Q),$$

```
if not (country == "한국" and province != "제주") :
        shipping_cost = 8000
```

```
if country != "한국" or province == "제주" :
        shipping_cost = 8000
```

중간점검

1. 다음의 조건에 해당하는 논리 연산식을 만들어 보시오. 변수는 적절하게 선언되어 있다고 가정한다.

   "나이는 25살 이상, 연봉은 3500만원 이상"

1. 수식 **not True**의 값은?

Lab: 물의 상태 출력하기

- 사용자로부터 온도를 입력받아서 현재 물의 상태를 출력하는 프로그램을 작성해보자.

```
온도를 입력하시오: 30
물의 상태는 액체입니다.
```

p133_water_state.py

```
##
# 이 프로그램은 온도에 따른 물의 상태를 출력한다.
#
temp = float(input("온도를 입력하시오: "))

if temp <= 0 :
        print("물의 상태는 얼음입니다.")
elif temp > 0 and temp < 100 :          # 논리 연산자를 사용한다.
        print("물의 상태는 액체입니다.")
else :
        print("물의 상태는 기체입니다.")
```

Lab: 동전 던지기 게임

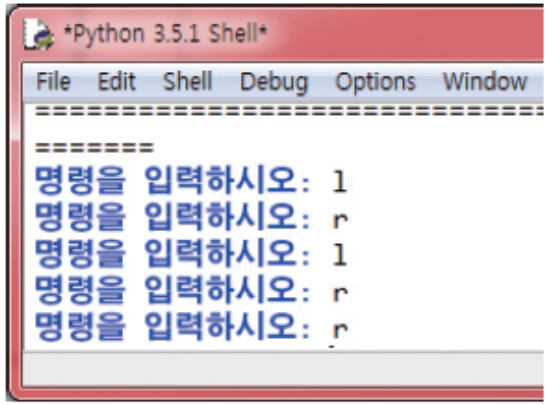
```
동전 던지기 게임을 시작합니다.
뒷면입니다.
게임이 종료되었습니다.
```

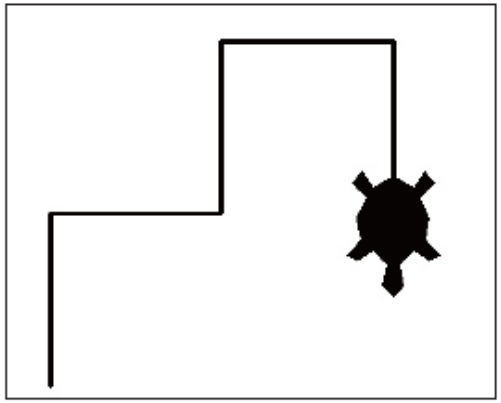
p134_coin_tossing.py

```
import random

print("동전 던지기 게임을 시작합니다.")

coin = random.randrange(2)

if coin == 0 :
        print("앞면입니다.")
else :
        print("뒷면입니다.")

print("게임이 종료되었습니다.")
```

Lab: 거북이 제어하기

- 파이썬 쉘에서 “l”을 입력하면 거북이가 왼쪽으로 100픽셀 이동하고 “r”을 입력하면 거북이가 오른쪽으로 100픽셀 이동하는 프로그램





p135_control_turtle.py

```
import turtle

# 거북이를 만든다.
t = turtle.Turtle()

# 커서의 모양을 거북이로 한다.
t.shape("turtle")

# 거북이가 그리는 선의 두께를 3으로 한다.
t.width(3)

# 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)
    if command == "q":                  # 사용자가 “q“을 입력하였으면
        break                           # 무한 루프를 빠져나간다.

turtle.mainloop()
turtle.bye()
```

중첩 if 문

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

- 배송지가 한국이면 다음과 같이 배송비가 결정된다. - ”상품의 가격이 2만원 이상이면 배송비는 없고 그렇지 않으면 3000원의 배송비가 붙는다.“

- 배송지가 미국이면 다음과 같이 배송비가 결정된다. - ”상품의 가격이 10만원 이상이면 배송비는 없고 그렇지 않으면 8000원의 배송비가 붙는다.“

배송비 계산 프로그램

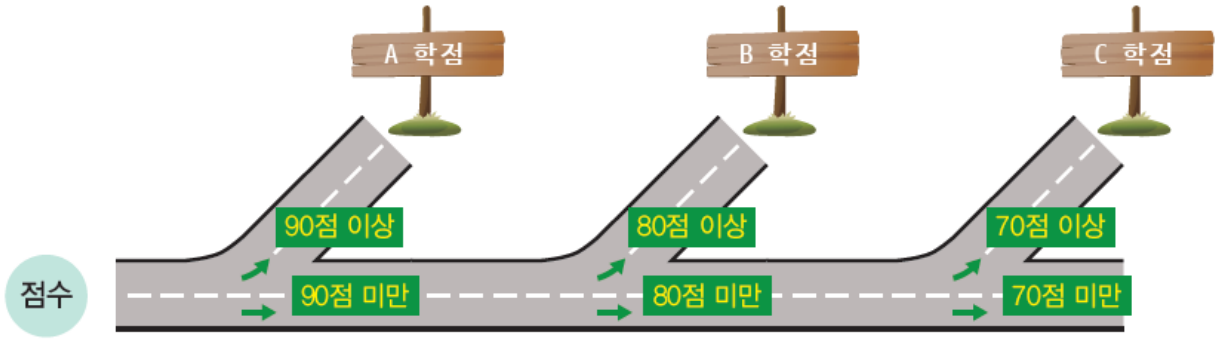
p137_shipping.py

```
# 사용자로부터 상품의 가격을 입력받는다.
country = input("배송지(현재는 korea와 us만 가능): ")
price = int(input("상품의 가격: "))

# 배송비를 결정한다.
if  country == "korea" :
        if  price >= 20000 :
                shipping_cost = 0
        else :
                shipping_cost = 3000
else :
        if  price >= 100000 :
                shipping_cost = 0
        else :
                shipping_cost = 8000

# 배송비를 출력한다.
print("배송비 = ", shipping_cost)
```

```
배송지(현재는 korea와 us만 가능): us
상품의 가격: 120000
배송비 = 0
```

연속 if 문

- 조건에 따라 다중으로 분기되는 결정을 내려야 하는 경우.



- 연속 if문에서는 순서가 아주 중요하다.
- 독립적인 if문을 여러 개 쓰는 것과 차이점을 알아야 한다.

학점 결정 예제

p138_if.py

```
# 1 올바른 조건 순서

score = 72

if score >= 90 :
        print("학점 A")
elif score >= 80 :
        print("학점 B")
elif score >= 70 :
        print("학점 C")
elif score >= 60 :
        print("학점 D")
else :
        print("학점 F")
```

학점 C

```
# 2 잘못된 조건 순서

score = 72

if score >= 60 :
        print("학점 D")
elif score >= 70 :
        print("학점 C")
elif score >= 80 :
        print("학점 B")
elif score >= 90 :
        print("학점 A")
else :
        print("학점 F")
```

학점 D

```
# 3 독립적인 if을 여러개 사용

score = 72

if score >= 90 :
        print("학점 A")
if score >= 80 :
        print("학점 B")
if score >= 70 :
        print("학점 C")
if score >= 60 :
        print("학점 D")
if score < 60 :
        print("학점 F")
```

학점 C
학점 D

Lab: 리히터 규모

- 사용자로부터 지진의 리히터 규모를 받아서 그 영향을 출력하는 프로그램을 작성

리히터 규모	영향
2.0 미만	지진계에 의해서만 탐지 가능합니다.
2.0–3.9	물건들이 흔들리거나 떨어집니다.
4.0–6.9	빈약한 건물에 큰 피해가 있습니다.
7.0–7.9	지표면에 균열이 발생합니다.
8.0–9.0	대부분의 구조물이 파괴됩니다.

Solution

p140_earthquake.py

```
##
#           이 프로그램은 리히터 규모를 받아서 피해정도를 출력한다.
#
scale = float(input("리히터 규모를 입력하시오: "))

if scale >= 8.0 :
            print("대부분의 구조물이 파괴됩니다. ")
elif scale >= 7.0 :
            print("지표면에 균열이 발생합니다.")
elif scale >= 4.0 :
            print("빈약한 건물에 큰 피해가 있습니다. ")
elif scale >= 2.0 :
            print("물건들이 흔들리거나 떨어집니다.")
else :
            print("지진계에 의해서만 탐지 가능합니다. ")
```

```
리히터 규모를 입력하시오: 5.2
빈약한 건물에 큰 피해가 있습니다.
```

Lab: 8-매직볼

- 조건문을 이용하여 오늘의 운세를 알려주는 프로그램


```
행운의 매직볼로 오늘의 운세를 출력합니다.
확실히 이루어집니다.
```

p141_magic_ball.py

```
##
#            이 프로그램은 오늘의 운세를 출력한다.
#
import random

print("행운의 매직볼로 오늘의 운세를 출력합니다. ")
answers = random.randint(1, 8)
if answers == 1:
    print("확실히 이루어집니다.")
elif answers == 2:
    print("좋아 보이네요")
elif answers == 3:
    print("믿으셔도 됩니다.")
elif answers == 4:
    print("저의 생각에는 no입니다.")
else:
    print("다시 질문해주세요.")
```

Lab: 사용자 입력 검증하기

- 사용자가 선택할 수 있는 메뉴를 1번부터 3번까지 출력하고 사용자가 입력한 값이 1부터 3 사이에 있는지를 if 문으로 검사.

Solution

```
========================
메뉴 1번: 치즈 버거
메뉴 2번: 치킨 버거
메뉴 3번: 불고기 버거
========================
메뉴를 선택하세요:5
잘못 입력하셨습니다.
```

p142_user_input.py

```
##
#           이 프로그램은 사용자의 입력을 검증한다.
#
print("========================")
print("메뉴 1번: 치즈 버거")
print("메뉴 2번: 치킨 버거")
print("메뉴 3번: 불고기 버거")
print("========================")

selection = int(input("메뉴를 선택하세요:"))

if selection >= 1 and selection <= 3 :
        print("메뉴 ", selection)
else :
        print("잘못 입력하셨습니다.")
```

Lab: 축구게임

- 난수를 이용하여 간단한 축구 게임을 작성하여 보자. 사용자가 컴퓨터를 상대로 페날티킥을 한다고 생각하자. 사용자는 다음의 3가지 영역 중에서 하나를 선택하여 페날티킥을 한다. 컴퓨터도 난수를 생성하여 3개의 영역 중 에서 하나를 수비한다.

Solution

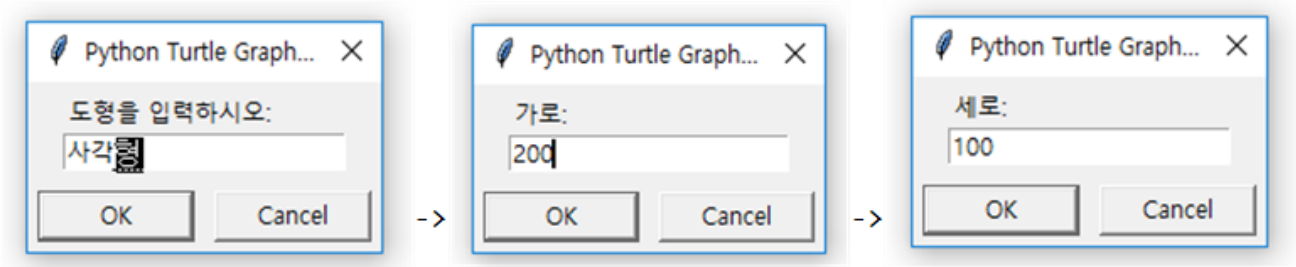
```
어디를 수비하시겠어요?(왼쪽: 1, 중앙: 2, 오른쪽: 3)1
페날티킥이 성공하였습니다.
```

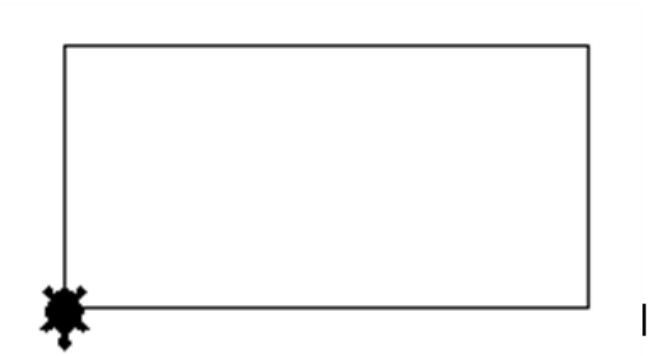
p143_soccer.py

```
##
#           이 프로그램은 축구 게임을 구현한다.
#
import random

computer_choice = random.randint(1, 3)
user_choice = input("어디를 수비하시겠어요?(왼쪽: 1, 중앙: 2, 오른쪽: 3)")
if computer_choice == user_choice:
        print("수비에 성공하셨습니다. ")
else:
        print("페날티킥이 성공하였습니다. ")
```

Lab: 도형 그리기

- 터틀 그래픽을 이용하여 사용자가 선택하는 도형을 화면에 그리는 프로그램을 작성해보자. 도형은 “사각형”, “삼각형”, “원” 중의 하나이다.



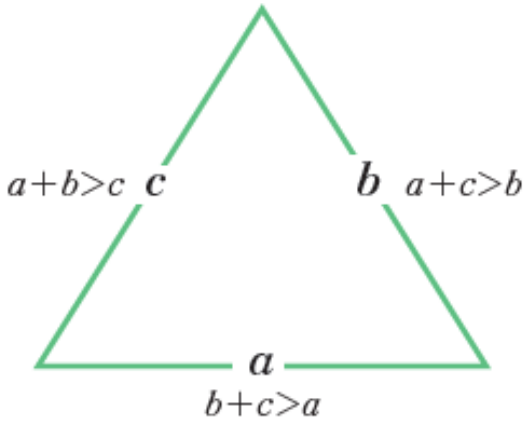


```
##                                                                    p144_draw_rect.py
#          이 프로그램은 사용자가 원하는 도형을 화면에 그린다.
#
import turtle
t = turtle.Turtle()
t.shape("turtle")

s = turtle.textinput("", "도형을 입력하시오: ")

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

turtle.mainloop()
turtle.bye()
```

- 도전문제 : 위의 프로그램에서 '사각형' 만 지원하고 있다. '삼각형', '원'인 경우에 그리는 코드를 추가하라.

Lab: 올바른 삼각형 구별하기

- 삼각형의 각 변의 길이를 a, b, c라고 했을 때, 다음과 같은 수식이 성립되면 올바른 삼각형이다.



- 사용자로부터 삼각형 변의 길이를 받아서 유효한 삼각형인지를 검사하는 프로그램을 작성하라.

Solution

```
삼각형의 한 변을 입력하시오: 8
삼각형의 한 변을 입력하시오: 10
삼각형의 한 변을 입력하시오: 3

올바른 삼각형
```

p146_triangle.py

```
a = int(input("삼각형의 한 변을 입력하시오: "))
b = int(input("삼각형의 한 변을 입력하시오: "))
c = int(input("삼각형의 한 변을 입력하시오: "))
if (a + b) > c and (b + c) > a and (a + c) > b :
        print("올바른 삼각형")
else :
        print("올바르지 않은 삼각형")
```

이번 장에서 배운 것

- 문장의 실행 순서를 바꾸는 2가지 종류의 제어문은 조건문과 반복문이다.
- if-else 문의 구조를 주석으로 설명하여 보시오.
- if(조건식)
- 문장1;
- //_____
- else
- 문장2;
- //_____

- 조건에 따라서 실행되어야 하는 문장이 두개 이상이면 이들 문장을 들여쓰기 한다. 이것을 복합문(블록)이라고 한다.
- if-else 문 안에 다른 if-else 문이 포함될 수 있다.

Q & A