

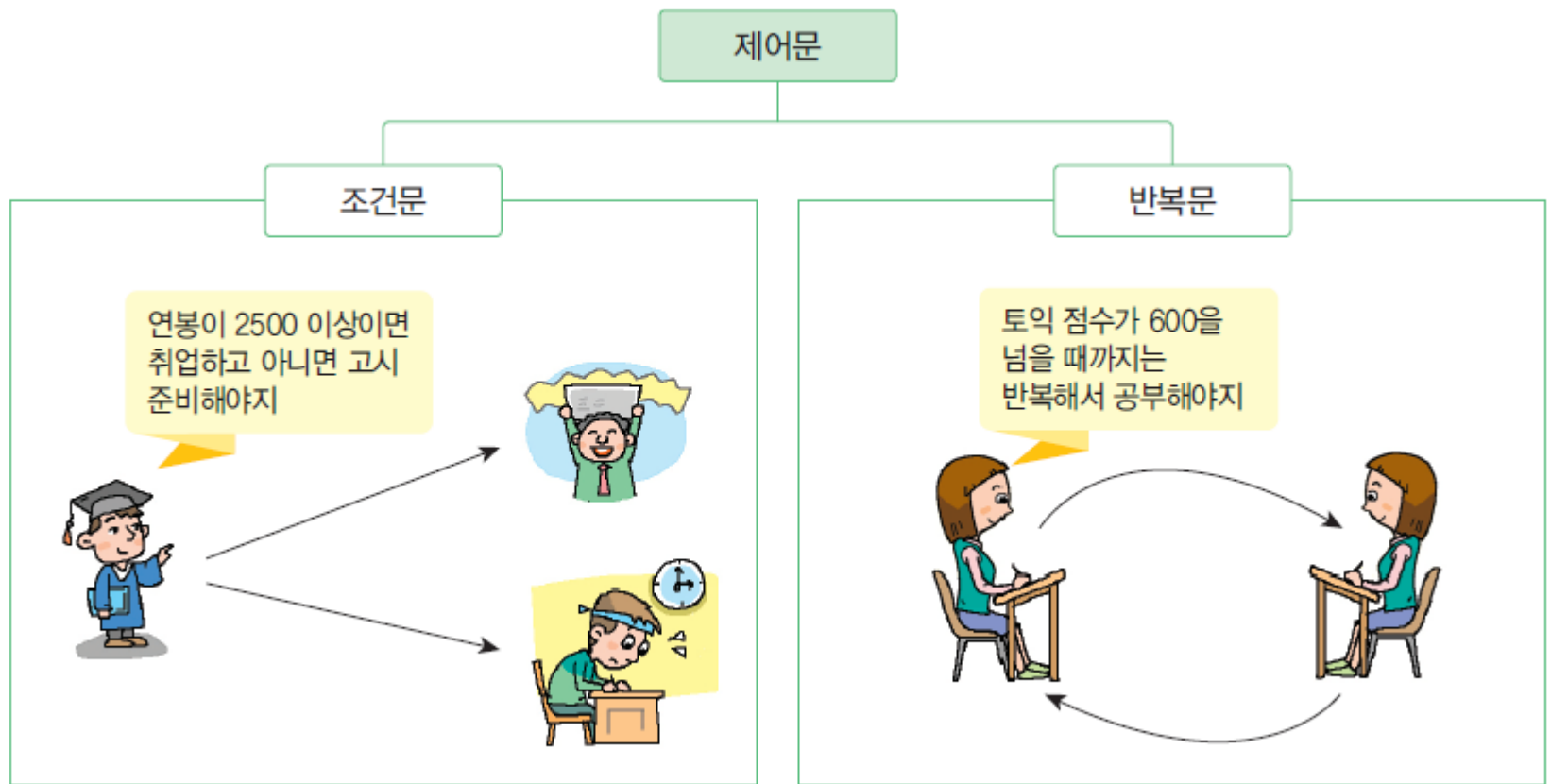
## 3장 조건문

# 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.

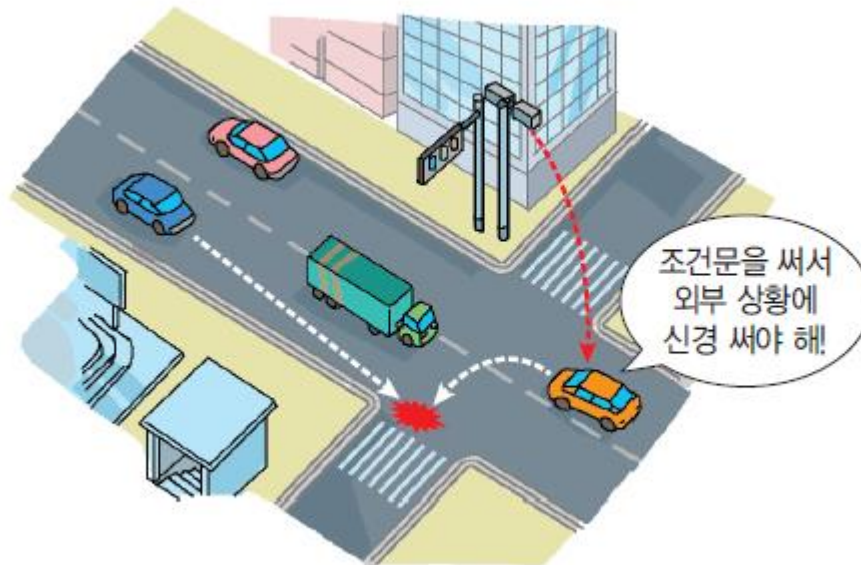


# 제어문

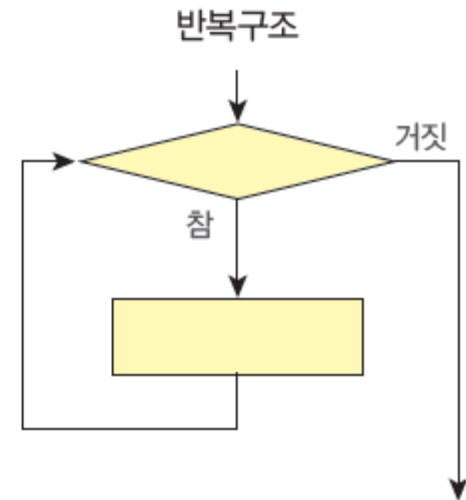
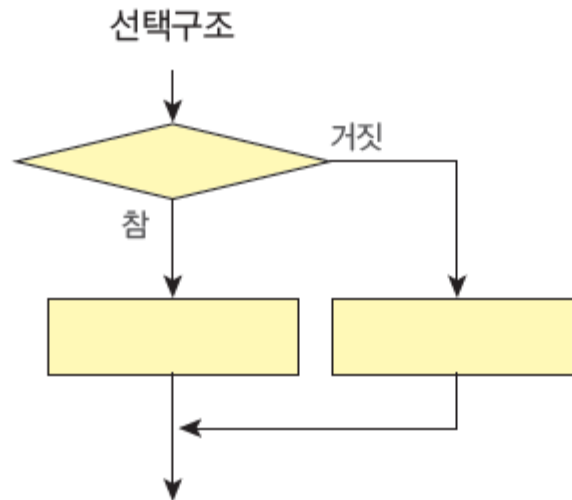
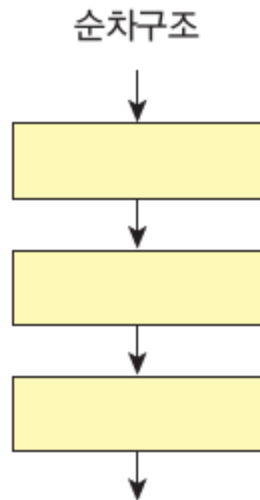


# 조건문의 중요성

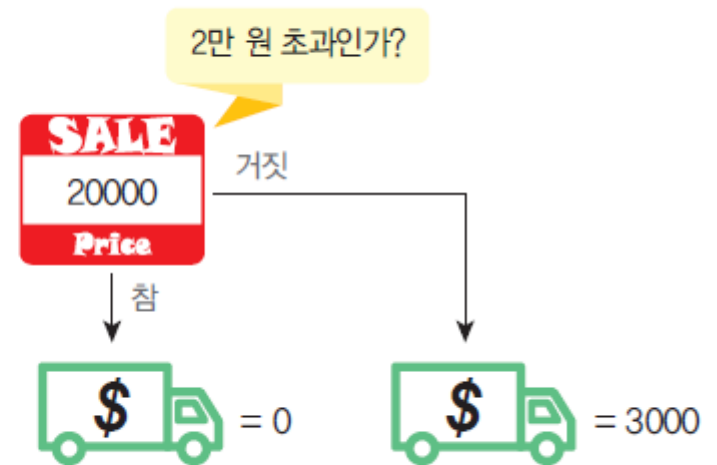
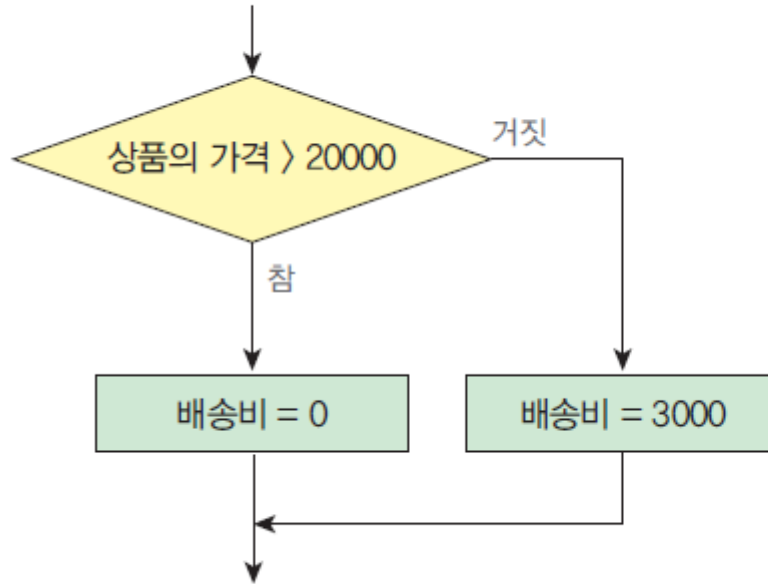
- 만약 프로그램에 조건문이 없다면 프로그램은 항상 동일한 동작만을 되풀이 할 것이다.



# 3가지의 제어구조



# if-else 문



# if-else 문

## 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

else 절은 생략될 수도 있다.

if와 else는 같은 위치여야 한다.

# 배송비 계산 프로그램

```
# 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

```
if price > 20000 :
```

```
    shipping_cost = 0  
    discount = 0.1
```

블록: 여러 문장들을 묶은 것이다.

```
else :
```

```
    shipping_cost = 3000
```

```
if price > 20000 :
```

```
    shipping_cost = 0
```

```
    discount = 0.1
```

블록

```
else :
```

```
    shipping_cost = 3000
```

# else는 없을 수도 있다.

<pre>shipping_cost = 3000 if price &gt; 20000 :     shipping_cost = 0</pre>	<pre># 기본적으로 배송비는 3000원이다. # 만약 상품의 가격이 2만원 초과이면 # 배송비가 없다.</pre>
---	---

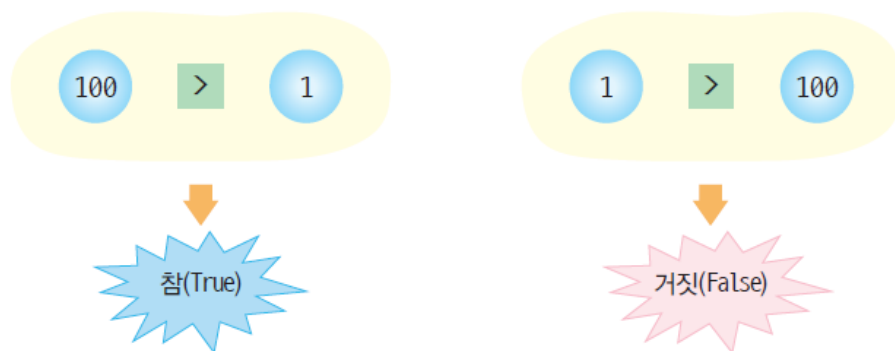
파이썬에서는 들여쓰기가 아주 중요하다. if-else 문에서도 들여쓰기가 잘못되면 오류가 발생한다.

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

0      1      들여쓰기 레벨

# 관계 연산자

연산	의미	수학적 표기
$x == y$	x와 y가 같은가?	$=$
$x != y$	x와 y가 다른가?	$\neq$
$x > y$	x가 y보다 큰가?	$>$
$x < y$	x가 y보다 작은가?	$<$
$x \geq y$	x가 y보다 크거나 같은가?	$\geq$
$x \leq y$	x가 y보다 작거나 같은가?	$\leq$



# 부울 변수

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

True

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

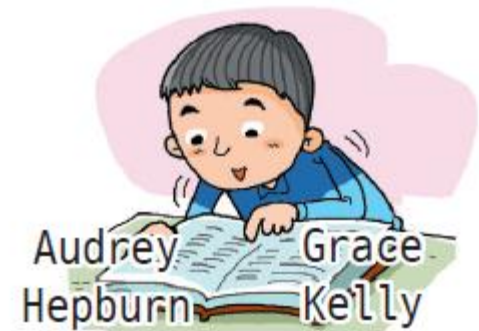
# 문자열 비교

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

True

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

True



# 실수 비교

```
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. ")
```

$\text{sqrt}(3.0) * \text{sqrt}(3.0)$ 은 3.0과 같지 않다.

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

# Lab: 산술 퀴즈 프로그램

- 초등학생들을 위하여 산수 퀴즈를 발생시키는 프로그램을 작성해보자.

25 + 78 = 103  
True

25 + 78 = 100  
False

# Solution

```
##  
#           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)
```



덧셈 뿐만 아니라 뺄셈 문제도 출제할 수 있도록 위의 프로그램을 수정하라.



# 조건 연산자

참 거짓

max\_value = (x if x > y else y)

```
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
```

# 조건 연산자 예제

```
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: 산술 퀴즈 프로그램

- 사용자로부터 정수를 입력받아서 짝수인지 홀수인지를 검사하는 프로그램을 작성해보자.

정수를 입력하시오: 10  
짝수입니다.

# Solution

```
number = int(input("정 수를 입력하시오: "))
```

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

```
else:  
    print("홀수입니다.")
```



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  
10%에서 사은품을 받아주세요.  
할인된 가격 = 170.0

정가를 입력하시오: 80  
할인된 가격 = 72.0

# Solution

```
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

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



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

# 논리 연산자

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

`price > 20000` *and* `card == "python"`

가격이 2만원 초과

그리고

파이썬 카드이면



# 예제

```
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.
```

# Lab: 동전 던지기 게임

- 동전을 던지기 게임을 작성해보자.

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

## Solution

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

# 중첩 if 문

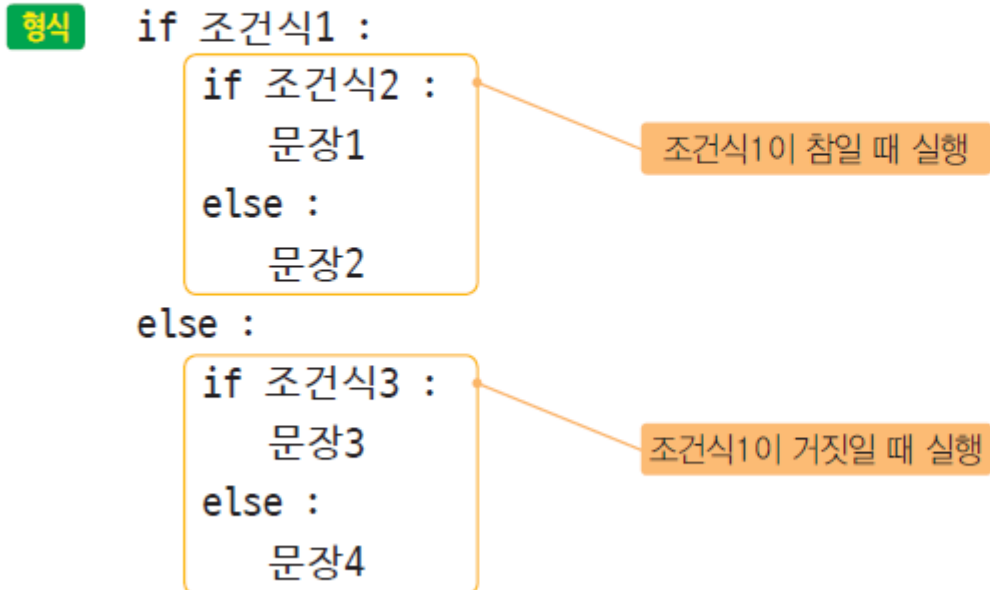
- if 문 안에 다른 if 문이 들어갈 수도 있다. 이것을 중첩 if 문이라고 한다.

## Syntax: 중첩 if 문

**형식** if 조건식1 :  
    if 조건식2 :  
        문장1  
    else :  
        문장2  
else :  
    if 조건식3 :  
        문장3  
    else :  
        문장4

조건식1이 참일 때 실행

조건식1이 거짓일 때 실행

The diagram illustrates the syntax of a nested if statement. It shows an outer 'if' block with a condition '조건식1'. Inside this block, there is an inner 'if' block with condition '조건식2' and an 'else' branch. The inner 'if' block has two branches: '문장1' and '문장2'. The outer 'if' block also has an 'else' branch containing another 'if' block with condition '조건식3' and an 'else' branch with '문장3' and '문장4'. Two callout boxes with orange borders point to the inner 'if' blocks. The first callout, labeled '조건식1이 참일 때 실행', points to the inner 'if' block with condition '조건식2'. The second callout, labeled '조건식1이 거짓일 때 실행', points to the inner 'if' block with condition '조건식3'.

# 배송비 계산 프로그램

- 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."

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

# 배송비 계산 프로그램

```
# 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)
```

# 연속 if 문

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

정수를 입력하시오: 10  
양수입니다.

if 조건:  
 문장1  
elif 조건:  
 문장 2  
...  
elif 조건:  
 문장 3  
else :  
 문장 n

if 조건:  
 문장1  
elif 조건:  
 문장 2  
...  
elif 조건:  
 문장 3

# 학점 결정 예제

```
# 성적을 받아서 학점을 결정하는 프로그램

score = int(input("성적을 입력하시오: "))

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")
```

```
성적을 입력하시오: 88
학점 B
```

# 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.

리히터 규모를 입력하시오: 5.2  
빈약한 건물에 큰 피해가 있습니다.



# Solution

```
##  
#         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. ")
```

# Lab: 8 매직볼



- 조건문을 이용하여서 오늘의 운세를 알려주는 프로그램을 개발해보자.



해운의 매지볼로 오늘의 운세를 출력합니다.  
확실히 이루어집니다.

# Solution

```
##  
#           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 검사해보자.



=====

메뉴 1번: 치즈 버거

메뉴 2번: 치킨 버거

메뉴 3번: 불고기 버거

=====

메뉴를 선택하세요:5

잘못 입력하셨습니다.

# Solution

```
##  
#           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.")
```

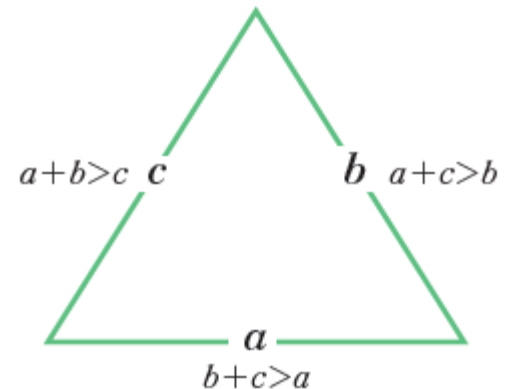
# Lab: 올바른 삼각형 구분



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

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

올바른 삼각형



## Solution

```
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")
```