

따라하며 배우는

파이썬과 데이터 과학



4장

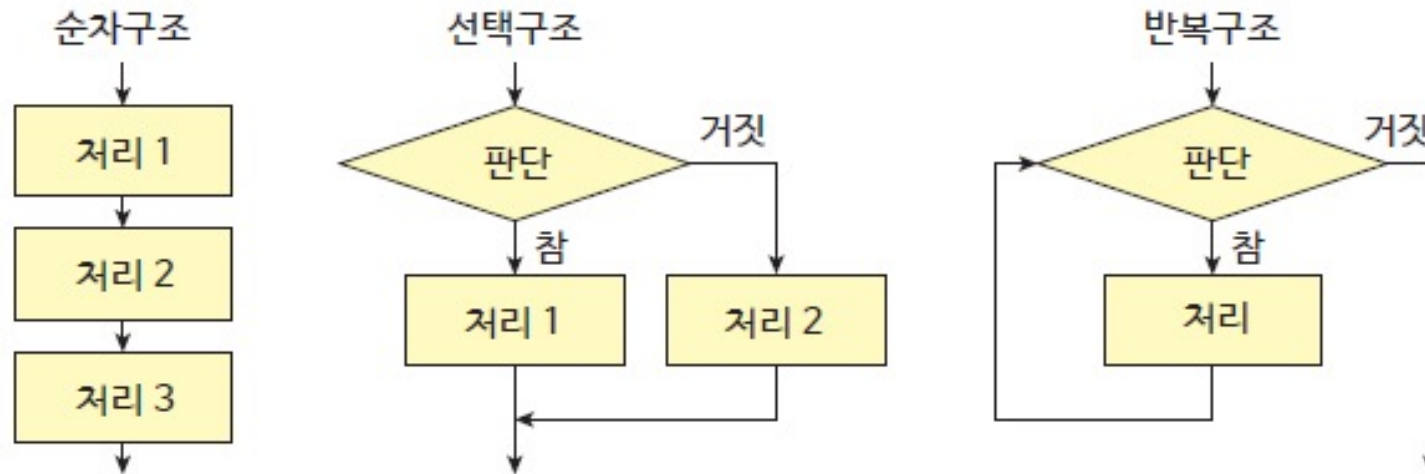
조건을 따져 실행해보자

이장에서 배울 것들

- 순차적 실행이 아닌 조건에 따라 수행되는 제어문에 대하여 이해해 보아요.
- if-else 문을 이해하고 사용할 수 있게 될 것이에요.
- 관계연산자와 논리연산자를 학습해 보아요.
- 블록의 개념을 이해하고 활용할 수 있게 될 것이에요.
- 중첩 if-else 문을 학습해 보아요.
- 조건에 따라 다르게 동작하는 프로그램을 작성해 보아요.
- 좀더 동적으로 동작하는 프로그램을 작성할 수 있게 될 것이에요.

4.1 프로그램의 기본 제어 구조는 크게 세 가지가 있다.

- **순차 구조**sequence - 여러 명령이 순차적으로 실행되는 구조이다.
- **선택 구조**selection - 여러 개 중 하나의 명령문을 선택하여 실행하는 구조이다.
- **반복 구조**iteration - 동일한 명령이 반복되면서 실행되는 구조이다.
- 위의 세 구조를 표현하는 **순서도**flowchart



4.1 프로그램의 기본 제어 구조는 크게 세 가지가 있다.

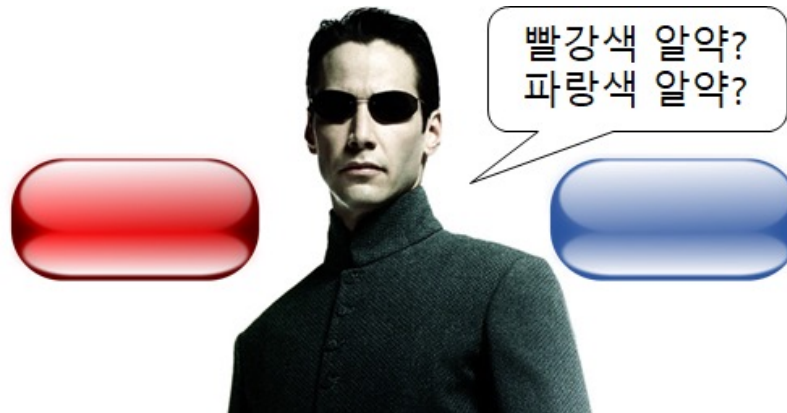


- 이것들은 레고의 기본 블록과 유사하다. 레고의 거의 모든 작품은 기본 블록 몇 가지만을 이용하여 만들어진다.
- 프로그램도 마찬가지이다. 어떠한 프로그램이라도 3가지의 기본 블록만 있으면 만들 수 있다.
- 프로그램의 기본 블록을 쉽게 이해하려면 이것을 자동차가 주행하는 도로로 생각하면 된다.

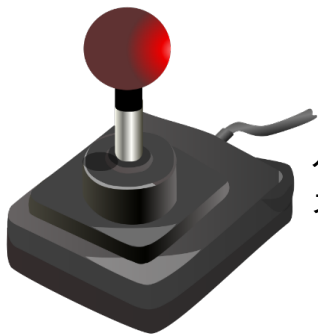
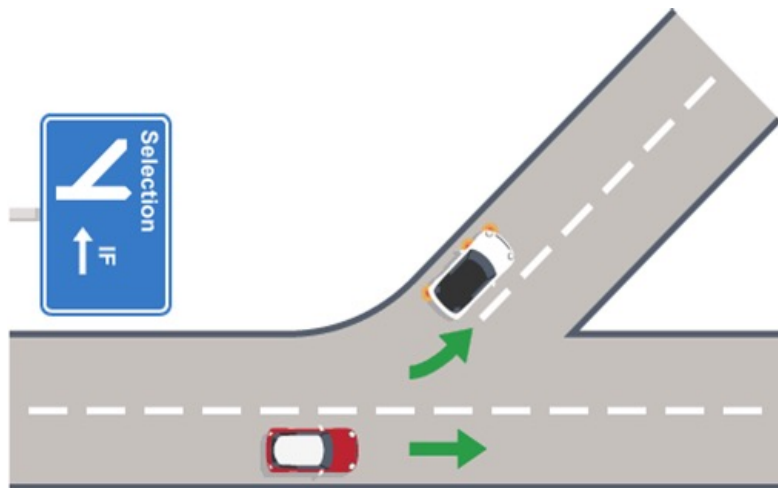
- 순차 구조는 자동차가 직진하는 도로라고 생각할 수 있다.
- 선택 구조는 왼쪽과 같이 자동차가 2가지의 길 중에서 하나를 선택하여 주행하는 교차로이다. 반복 구조는 자동차가 회전하면서 주행하는 회전 교차로라고 할 수 있다.

4. 2 왜 선택 구조가 필요한가?

- 프로그램의 어떤 단계에서는 진행할 수 있는 경로가 하나 이상인 경우가 있는데, 이때 우리는 어떤 경로를 선택할 것인지를 결정하여야 한다.
- 만약 선택 구조가 없다면 프로그램은 항상 동일한 동작만을 되풀이 할 것이다. 프로그램이 항상 동일한 동작만 한다면 언제나 정해진 결론에 도달할 것이다.



4. 2 왜 선택 구조가 필요한가?



선택에 의해서 가는
길이 달라짐

- 미성년자가 아니라면 영화 "킹덤"을 시청할 수 있다.
- 게임에서 철수가 정답을 맞혔으면 철수의 점수를 1만큼 증가한다.
- 게임 사용자가 외계인 우주선을 맞추었으면 폭발 사운드를 출력한다.
- 파일이 하드 디스크에 없으면 오류 메시지를 출력한다.
- 11:59분이 지나면 수강신청을 할 수 있다(특정 시간에 따른 실행).

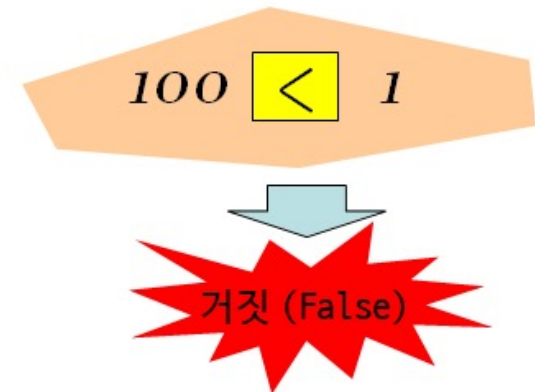
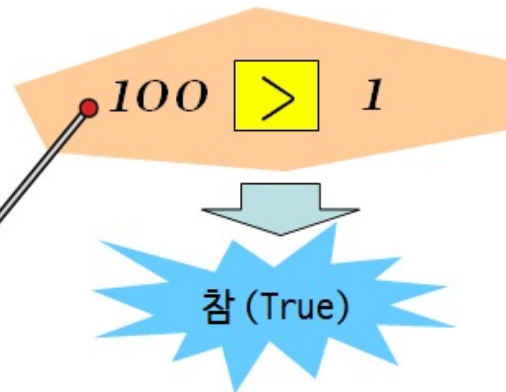
4. 3 조건이 맞을 때만 실행되는 if 문

- 성적이 60점 이상이면 '합격입니다'를 출력하고, 60점 미만이면 '불합격입니다'를 출력한다고 하자. 이것은 성적이라는 데이터가 60점 이상인지 그렇지 않은지를 판단해야 한다.
- 이렇게 어떤 **조건**condition을 만족하는지 그렇지 않은지를 판정하는 식을 **조건식**condition expression이라고 한다. 그리고 이 조건식은 참 또는 거짓의 값을 갖는 **부울**bool형으로 평가된다.
- 앞에서 배운 **관계 연산자**relational operator는 두 개의 피연산자를 비교하는데 사용되는데, 관계 연산자의 결과는 **참**True 아니면 **거짓**False으로 나타난다. 따라서 이것은 참과 거짓의 값을 갖는 조건식이 될 수 있다.

4. 3 조건이 맞을 때만 실행되는 if 문

- 파이썬에서 참과 거짓은 True와 False로 표시되기 때문에 True 또는 False가 관계 연산의 값으로 생성된다. **100 > 1** 라는 관계식을 예로 들어보자.
- 100이 1보다 크기 때문에 이 수식은 참을 의미하는 값 True를 생성한다. 반면 **1 > 100** 수식은 거짓이므로 False 값이 생성된다. 조건식은 이렇게 참과 거짓의 값을 가질 수 있는 수식을 의미한다.

조건 연산자는 피연산자들의 크기를 비교하여 True, False를 돌려줘요!



4. 3 조건이 맞을 때만 실행되는 if 문

- 실제 코드로 확인해 보자. 다음과 같이 변수 x에 100을 넣어준 후 아래와 같이 if 문을 작성해 보자. 이 경우 조건식에 해당하는 $x > 1$ 이 True를 반환하므로 실행할 내용이 선택되어 실행된다.
- 따라서 화면에는 'x는 1보다 큼니다.'가 출력되는 것을 볼 수 있다. 이때 :(콜론) 아래의 줄은 **반드시 들여쓰기**를 해야만 한다

```
x = 100
if x > 1:
    print("x는 1보다 큼니다.")
```

x는 1보다 큼니다.

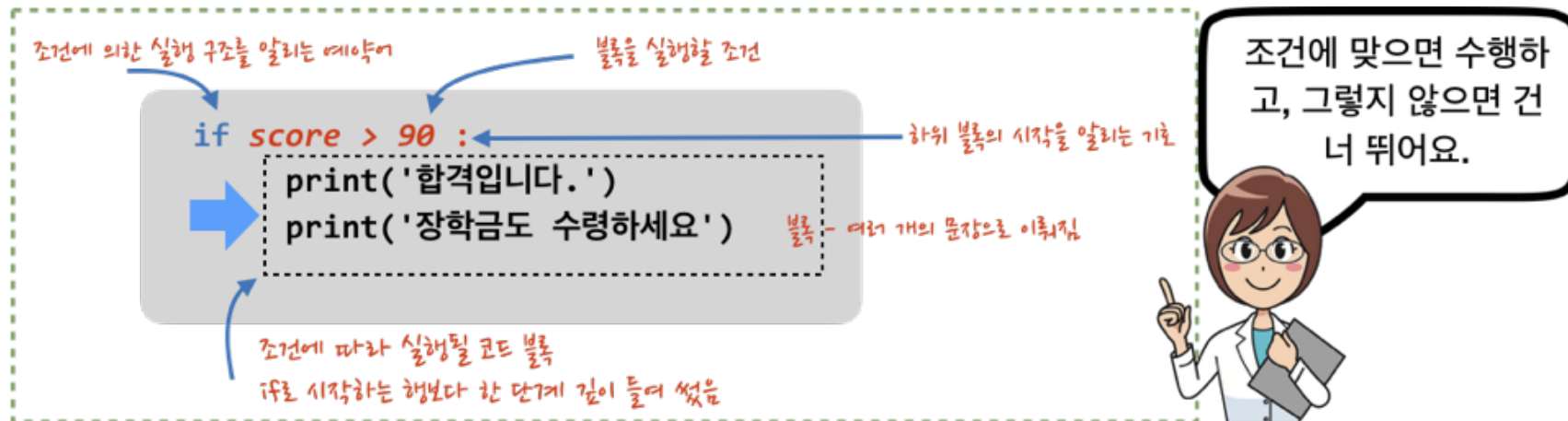


잠깐 - 대입 연산자 =와 관계 연산자 ==는 다른 연산자이다.

반복적으로 언급하는 내용이지만, 코딩을 처음 시작하는 사람들이 자주 실수하는 부분이다. = 연산자는 오른쪽의 값을 왼쪽 변수에 담는 연산이고, == 연산자는 좌우의 값이 같은지를 보고 참True과 거짓False 중 하나를 반환하는 연산자이다.

4. 4 들여쓰기가 아주 중요한 파이썬 : 블록은 들여쓰기로 완성

- 성적이 90점 이상이면 합격과 동시에 장학금도 받을 수 있다고 출력하려면 어떻게 해야 할까? 이런 경우에는 다음과 같이 들여쓰기를 이용하여서 문장들을 묶을 수 있다.
- 아래의 코드에서 score의 값이 90이상이면 print()를 호출하는 2개의 문장이 실행된다. 이들 문장들이 동일한 개수의 공백을 가지고 있다는 것에 유의하라. 이들 모두는 동일한 **블록**block에 속해 있다.



4.4 들여쓰기가 아주 중요한 파이썬 : 블록은 들여쓰기로 완성

- 하나의 블록에 속하는 문장들은 모두 같이 실행된다. 블록에 있는 문장들은 그 위에 있는 문장들과 비교할 때, 앞에 4칸의 들여쓰기 공백을 두고 있다. 이 공백들을 □로 표시해보면 다음과 같다.

```
score = int(input("점수를 입력하시오: "))
if score >= 90 :           # 조건이 참일 경우 들여쓰기 블록이 모두 실행됨
    □□□□print("축하합니다.")
    □□□□print("합격입니다.")
    □□□□print("장학금도 받을 수 있습니다.")
```

- 파이썬 에서 문장 앞에 동일한 개수의 공백이 있으면 이들 문장들은 하나의 블록에 속하게 된다. 만약 실수로 공백을 더 많이 추가하였다면 오류가 발생한다.

```
score = int(input("점수를 입력하시오: "))
if score >= 90 :           # 주의 : 들여쓰기의 크기가 다를 경우 오류가 출력됨
    □print("축하합니다.")
    □□print("합격입니다.")
    □□□print("장학금도 받을 수 있습니다.")
```

주의 : 들여쓰기의 크기가 달라
지면 오류!!

SyntaxError: multiple statements found while compiling a single statement

4. 4 들여쓰기가 아주 중요한 파이썬 : 블록은 들여쓰기로 완성



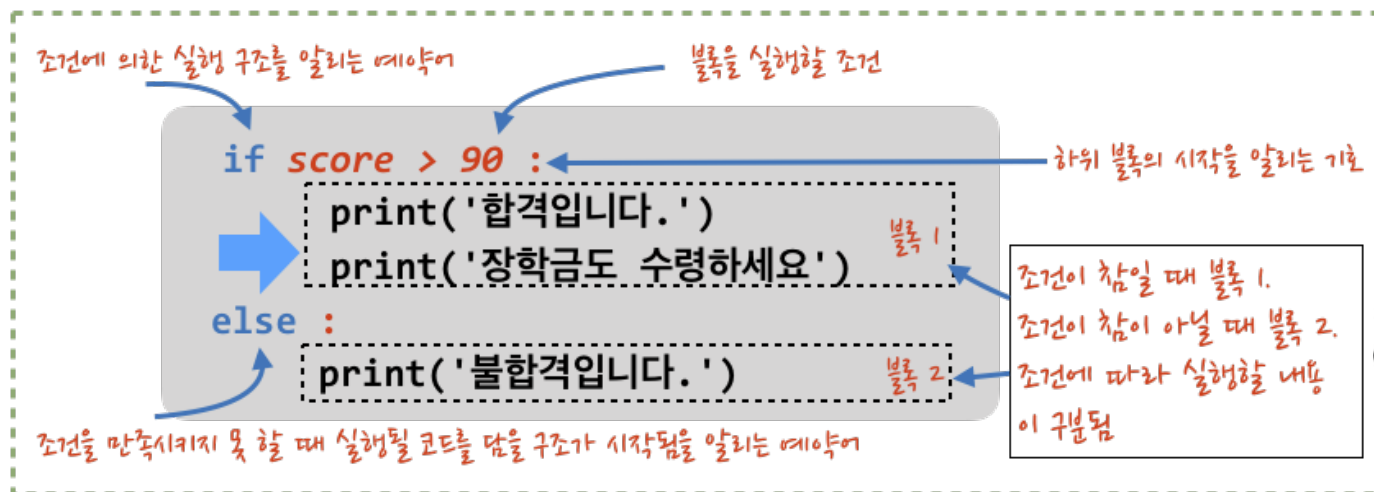
도전문제 4.1

어린이 보호구역내의 과속단속 카메라로부터 다음과 같이 자동차의 속도를 km/h 값으로 입력 받으시오. 이 값이 30 이상이면 다음과 같이 "과속입니다. 속도를 줄이세요."를 출력하여라.

속도를 입력하세요 : 33
과속입니다. 속도를 줄이세요.

4. 5 배타적 조건에 따라 실행하는 if-else 문

- 파이썬에서는 선택 구조를 위하여 if-else 문을 제공한다. 성적이 60점 이상이면 합격으로, 60점 미만이면 불합격으로 처리해야 한다고 하자. 파이썬으로 다음과 같이 작성할 수 있다.
- if-else 문장은 "만약 조건이 참이면 이것을 실행하고, 조건이 참이 아니라면 저것을 실행해!"라고 말하는 것과 같다. if-else 문에서는 조건을 수식으로 표현하며, 그 수식을 바로 '조건식' 이라고 한다.



조건에 따라 수행할
내용이 두 종류이네요.



4. 5 배타적 조건에 따라 실행하는 if-else 문

- 조건식 뒤에는 콜론(:)이 있다. 콜론(:)은 파이썬 인터프리터에게 "아직 전체 문장을 끝나지 않았으니 잠시 해석을 미뤄 달라. 이 다음에는 이 문장에 딸린 코드 블록이 나타날 것이다."라고 요청하는 기호이다.

```
if score >= 60:
    print("합격입니다.")
else:
    print("불합격입니다.")
```

- 또 다른 예시로 24시 체계의 시간정보를 입력 받아 12보다 작으면 '오전입니다.' 12이상이면 '오후입니다.'를 출력하는 시계가 있다고 가정하자. 0에서 23까지의 시간이 입력될 경우 오전이면서 동시에 오후가 될 수 없으므로 이 조건은 **배타적 조건** exclusive condition이 되어 다음과 같은 if-else 문으로 표현 가능하다.

```
hour = 10

if hour < 12:
    print('오전입니다.')
else:
    print('오후입니다.')
```

4. 5 배타적 조건에 따라 실행하는 if-else 문



도전문제 4.2

어린이 보호구역내의 과속단속 카메라로 부터 다음과 같이 자동차의 속도를 입력 받으시오. 이 값이 30 이상이면 "과속입니다. 속도를 줄이세요."를 출력하고, 그렇지 않을 경우 "안전운전하세요."를 출력하여라.

속도를 입력하세요 : 28
안전운전하세요.



4. 6 if-else 문으로 다양한 코드를 작성해보자

- 사용자로부터 성적을 입력받아서 합격 여부를 판단해 보자. 아래 코드처럼 사용자의 성적을 입력받는다. 입력된 값이 60 이상이면 프로그램은 '합격입니다.'를 출력하고, 그렇지 않으면 '불합격입니다.'를 출력한다.

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

```
성적을 입력하시오: 80  
합격입니다.
```



4. 6 if-else 문으로 다양한 코드를 작성해보자

- 사용자로부터 정수를 입력 받아서 짝수인지 홀수인지를 검사하는 프로그램을 작성해보자.

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

양의 정수를 입력하시오: 10
짝수입니다.

- **짝수** : 2로 나누어떨어지는 수

예 2, 4, 6, 8, 10,

- **홀수** : 2로 나누어떨어지지 않는 수

예 1, 3, 5, 7, 9,

4. 6 if-else 문으로 다양한 코드를 작성해보자

- 만일 임의의 정수를 입력 받은 후 이 검사를 0 이상인 수에 대해서만 하고 음수인 경우에 대해서는 '음수입니다'를 출력하는 **좀 더 복잡한 조건을 설정**하려면 어떻게 할까?
- 이 경우에는 다음처럼 if-else문 내에 또다시 if-else 조건을 달아서 해결할 수 있다. 이러한 경우 조건문 내의 조건문을 내부 조건문 이라고 한다.

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

```
if num < 0 :  
    print("음수입니다.")  
else:  
    print("양수입니다.")  
    if num % 2 == 0 :  
        print("짝수입니다.")  
    else:  
        print("홀수입니다.")
```

음수 vs 양수

짝수 vs 홀수

```
정수를 입력하시오: 10  
양수입니다.  
짝수입니다.
```

4. 6 if-else 문으로 다양한 코드를 작성해보자



도전문제 4.3

- (1) 사용자로부터 정수를 입력받아 0보다 작으면 '음수입니다.'를 출력하고 그렇지 않을 경우 '음수가 아닙니다.'를 출력하는 프로그램을 작성하시오.
- (2) 게임 사용자의 게임 점수(game_score)를 입력받아서 10000점 이상이면 '고수입니다.'를 출력하고, 10000점 미만이면 '입문자입니다.'를 출력하는 프로그램을 **if-else** 문을 이용하여 작성하시오.
- (3) 두 정수를 입력받아 같을 경우 '두 값이 일치합니다.'를 출력하고 값이 다르면 '두 값이 일치하지 않습니다.'를 출력하는 프로그램을 **if** 문을 이용하여 작성하시오. 두 정수에 각각 100과 200이 할당되어 있는 경우와 300과 300이 할당되어 있는 경우의 출력문을 확인하시오.



LAB⁴⁻² 영화를 볼 수 있는 나이 제한 검사를 하자

15세 이상만 볼 수 있는 영화가 있다고 하자. 사용자의 나이를 물어 보면 영화를 볼 수 있는지 없는지 여부를 화면에 출력하는 프로그램을 작성해보자.



원하는 결과

나이를 입력하시오: 19
본 영화를 보실 수 있습니다.

나이를 입력하시오: 14
본 영화를 보실 수 없습니다.

LAB⁴⁻² 영화를 볼 수 있는 나이 제한 검사를 하자

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



도전문제 4.6

- 1) 15세 이상이면 "본 영화를 보실 수 있습니다." 메시지를 출력한 후 그 다음 줄에 "영화의 가격은 10000 원입니다."를 출력해보자. 만약 15세 미만이면 "영화를 보실 수 없습니다." 메시지에 추가로 "다른 영화를 보시겠어요?"를 출력해보자.
- 2) 교통 카드의 종류로 '청소년', '성인' 카드가 있다고 하자. 사용자에게 카드의 종류를 입력받아 청소년이면 '청소년입니다'를 출력하고 '성인'이면 '승인되었습니다'를 출력하는 **if-else** 조건문을 만들어서 출력해보자.

LAB⁴⁻⁴ 윤년 판단은 어떻게 하지

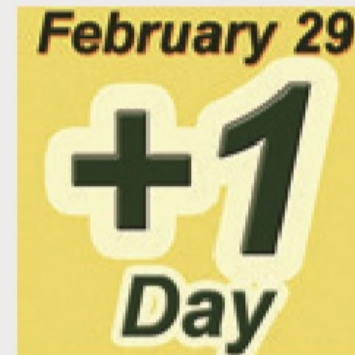
달력은 기본적으로 지구가 태양을 공전하는 시간을 기준으로 작성된다. 하지만 실제로 측정하여 보니 지구가 태양을 완전히 한 바퀴 도는데 걸리는 시간은 365일보다 약 1/4 만큼 더 걸린다. 따라서 매 4년마다 하루 정도 오차가 생기는 셈이다. 이것을 조정하기 위하여 윤년이 생겼다. 입력된 연도가 윤년인지 아닌지를 판단하는 프로그램을 만들어보자.

사용자로부터 연도를 입력받아서 다음과 같은 조건을 검사한다. 윤년은 다음의 조건을 만족해야 한다.

연도가 4로 나누어 떨어지면 윤년이다.

연수가 4로 나누어 떨어져도, 100으로는 나누어 떨어지지 않아야 한다.

연수가 400으로 나누어 떨어지는 해는 무조건 윤년으로 한다.



원하는 결과

연도를 입력하시오: 2012

2012 년은 윤년입니다.

LAB⁴⁻⁴ 윤년 판단은 어떻게 하지

```
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.")
```


LAB4-5 랜덤 함수로 동전 던지기 게임을 만들자

이 예제에서는 동전 던지기 게임을 작성해보자. 동전을 던지는 것은 난수를 생성하면 된다. 파이썬에서는 `import random`한 후에 `random.randrange(2)`과 같이 하면 0이나 1을 랜덤하게 생성할 수 있다. 아래의 코드에서 빈칸을 채워보자.

```
import random

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



원하는 결과

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

LAB⁴⁻⁵ 랜덤 함수로 동전 던지기 게임을 만들자

```
import random

print("동전 던지기 게임을 시작합니다.")
coin = random.randrange(2) # randrange(2)는 0 또는 1을 반환함
if coin == 0 :
    print("앞면입니다.")
else :
    print("뒷면입니다.")
print("게임이 종료되었습니다.")
```



도전문제 4.9

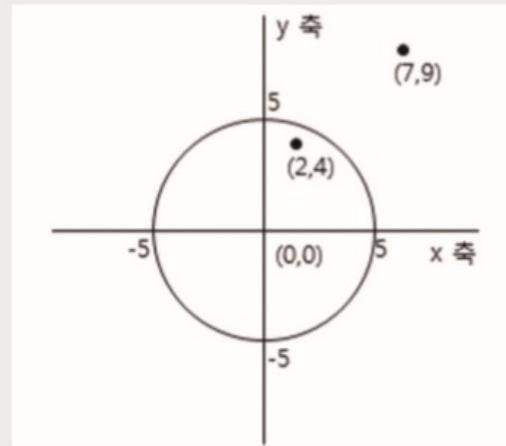
주사위 던지기 게임으로 변환해보자. `random.randint(1, 6)`을 사용하면 1에서 6까지의 정수를 랜덤하게 생성할 수 있다. 입력으로 '홍길동', '홍길순'과 같은 사용자의 이름을 받아서 홍길동의 주사위 번호, 홍길순의 주사위 번호를 출력한 후 이들을 비교하여 다음과 같이 출력하여라. 결과는 '홍길동이 이겼습니다', '홍길순이 이겼습니다', 혹은 '비겼습니다' 가운데 하나가 나타나도록 하여라.

Player1의 이름 : 홍길동
Player2의 이름 : 홍길순
.....주사위를 굴립니다.....
홍길동의 주사위 번호는 3
홍길순의 주사위 번호는 4
홍길순이 이겼습니다.



LAB⁴⁻⁶ 원의 내부에 있는 점일까 외부에 있는 점일까

중심이 원점 $(0,0)$ 에 있고 반지름이 5인 원이 있다고 가정하자. 사용자로부터 x 와 y 좌표를 입력받은 후, 입력받은 점의 좌표 (x, y) 가 원의 내부에 있으면 '원의 내부에 있음', 원의 외부에 있으면 '원의 외부에 있음'을 출력하는 다음과 같은 프로그램을 작성하시오. (힌트 : 어떤 점이 원 점과의 거리가 5보다 클 경우 원의 외부에 있으며, 5보다 작거나 같을 경우 원의 내부에 있다고 판단할 수 있다. 점 (x, y) 와 원점과의 거리는 $\sqrt{x^2 + y^2}$ 이다)



원하는 결과

점의 좌표 x, y 를 입력하시오 : 2 4
원의 내부에 있음
점의 좌표 x, y 를 입력하시오 : 7 9
원의 외부에 있음
점의 좌표 x, y 를 입력하시오 : 3 6
원의 외부에 있음

LAB⁴⁻⁶ 원의 내부에 있는 점일까 외부에 있는 점일까

```
x, y = map(float, input('점의 좌표 x, y를 입력하시오 : ').split())
if x*x + y*y > 5*5 :
    print('원의 외부에 있음')
else :
    print('원의 내부에 있음')
```

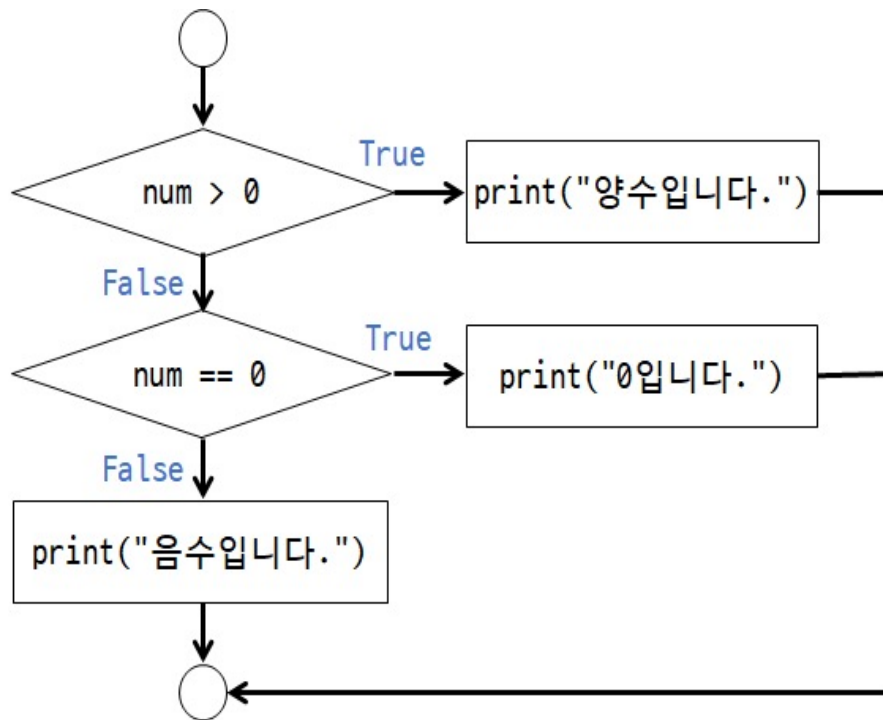


도전문제 4.10

위의 LAB은 원의 중심이 원점에 있는 경우를 가정하였다. 이제 원의 중심이 (3, 4)이고 원의 반지름이 10인 경우에 대하여 고려해보자. 다음과 같이 사용자로부터 x와 y좌표를 입력받은 후, 입력받은 점의 좌표 (x, y)가 원의 내부에 있으면 '원의 내부에 있음', 원의 외부에 있으면 '원의 외부에 있음'을 출력하는 다음과 같은 프로그램을 작성하시오.

점의 좌표 x, y를 입력하시오 : -9 9
원의 외부에 있음

4. 7 조건이 거짓일 때 연속하여 다른 조건을 검사



- if-else 문에서 조건이 거짓일 때 다른 조건을 검사할 수 있다. 이때는 elif 키워드를 사용하면 된다. elif는 "else if"를 합친 것이다.
- 사용자로부터 정수 num를 입력 받아서 이 값을 보고 "양수입니다.", "0입니다.", "음수입니다."를 출력하는 프로그램을 고려해보자.
- 우리가 첫 번째 조건문에서 True라는 값이 나올 경우 이를 실행하고 조건문에서 빠져나오게 되며, 그렇지 않을 경우(False)인 경우 두 번째 조건식을 검사한 후 이 조건에 따라서 결정을 내려야 하는 것이다.
- 이와 같은 구조의 프로그램은 if-elif-else를 사용하여 다음과 같이 구현할 수 있다.

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

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

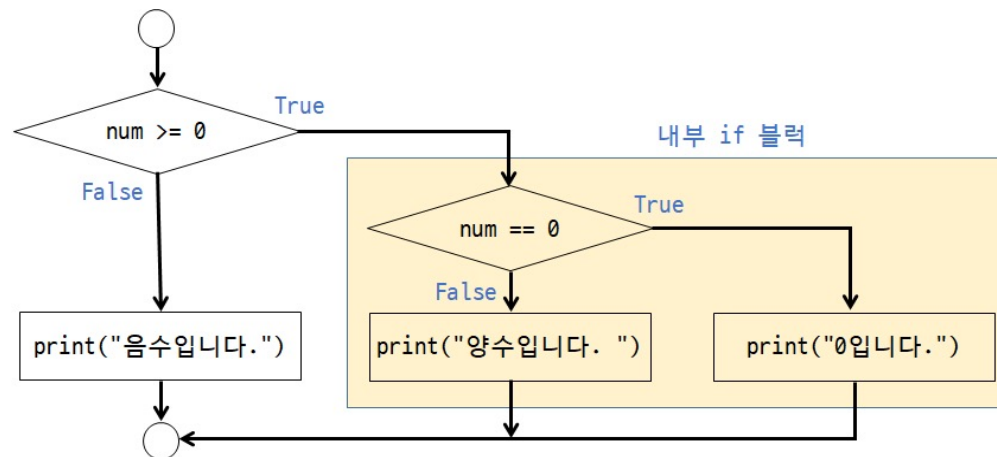
정수를 입력하시오: 0
0입니다.

정수를 입력하시오: 10
양수입니다.

정수를 입력하시오: -10
음수입니다.

4. 8 if-else 문 안에 if-else 문 넣기

- 필요에 따라 if 문 안에 다른 if 문이 들어갈 수도 있는데, 이것을 중첩 if 문이라고 한다. 이것은 다음과 같이 특정한 상황에서 True라는 조건이 나왔을 경우, 이 상황에 대해서 또 다른 세부적인 조건을 검사하는 상황에서 발생할 수 있다.
- 이 예제에서는 사용자로부터 num을 입력받은 후 이 값이 $\text{num} \geq 0$ 일 경우 num이 0인지 아닌지 또 다른 조건 검사를 하여 이에 따라 "0 입니다."와 "양수입니다."를 출력한다.



4. 8 if-else 문 안에 if-else 문 넣기

- 이제 다음과 같이 사용자에게서 정수를 받아서 양수인지, 0인지, 음수인지를 구별하여 화면에 출력하여 보자.

```
num = int(input("정수를 입력하시오: "))
if num >= 0:      # 반드시 들여쓰기를 하여 블록을 생성
    if num == 0:  # 블록 내에서 세부적인 조건을 한 번 더 검사
        print("0입니다.")
    else:
        print("양수입니다.")
else:
    print("음수입니다.")
```

정수를 입력하시오: 0
0입니다.

정수를 입력하시오: 10
양수입니다.

정수를 입력하시오: -10
음수입니다.



- 이와 같은 방법으로 if-elif-else 문과 동일한 기능을 하는 프로그램을 구현할 수 있으며 효율성과 가독성을 고려하여 개발자는 두 가지 문장중에서 적절한 문장을 선택할 수 있다.

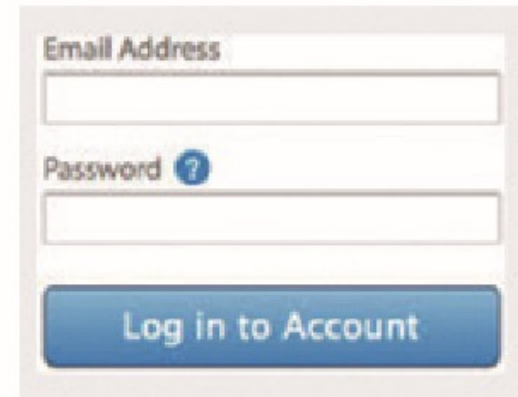
LAB⁴⁻⁷ 로그인 처리하기

사용자로부터 아이디를 받아서 프로그램에 저장된 아이디 'ilovepython'과 일치하는지 여부를 출력하는 프로그램을 작성해보자.

원하는 결과

아이디를 입력하시오: `ilovepython`
환영합니다.

아이디를 입력하시오: `iloveruby`
아이디를 찾을 수 없습니다.



Email Address

Password ?

Log in to Account

LAB⁴⁻⁷ 로그인 처리하기

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



도전문제 4.11

아이디 검사가 종료되면 바로 패스워드 검사를 하여 보자. 즉 다음과 같은 출력을 가지는 프로그램을 작성하며, 저장된 패스워드는 `mypass1234`라고 하자. 두 가지가 모두 맞을 경우 아래와 같이 "환영합니다."를 출력하도록 하고 그렇지 않을 경우 "아이디를 찾을 수 없습니다." 혹은 "비밀번호가 틀렸습니다."를 출력하도록 하자.

```
아이디를 입력하시오: ilovepython
패스워드를 입력하시오: mypass1234
환영합니다.
```

LAB⁴⁻⁸ 컴퓨터와 승부차기 게임을 만들어보자

난수를 이용하여 간단한 축구 게임을 작성하여 보자. 사용자가 컴퓨터를 상대로 페널티킥을 시도한다고 생각하자. 사용자는 다음의 '왼쪽', '중앙', '오른쪽'의 3가지 영역 중에서 하나를 선택하여 페널티킥을 한다. 컴퓨터도 난수를 생성하여 3개의 영역 중에서 하나를 수비한다. 출력은 다음과 같이 나타나도록 하자.

원하는 결과

어디를 공격하시겠어요?(왼쪽, 중앙, 오른쪽) : **왼쪽**
축하합니다!! 공격에 성공하였습니다.
컴퓨터의 수비위치 : **오른쪽**



LAB⁴⁻⁸ 컴퓨터와 승부차기 게임을 만들어보자

```
import random

n = random.randint(1, 3) # 랜덤하게 1, 2, 3 중 하나의 값을 생성
if n == 1:
    computer_choice = '왼쪽'
elif n == 2:
    computer_choice = '중앙'
else:
    computer_choice = '오른쪽'

user_choice = input('어디를 공격하시겠어요?(왼쪽, 중앙, 오른쪽) : ')
if computer_choice == user_choice:
    print('공격에 실패하셨습니다.')
else :
    print('축하합니다!! 공격에 성공하였습니다.')
print('컴퓨터의 수비위치 :', computer_choice)
```



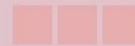
도전문제 4.12

이 문제를 풀기 위해 `options=['왼쪽', '중앙', '오른쪽']` 리스트를 생성하자. 다음으로 `random` 모듈을 사용하여 `computer_choice = random.choice(options)`와 같이하여 리스트에서 임의의 문자열을 추출하는 방식으로 다시 작성해 보자. 리스트에 대한 상세한 설명은 6장에서 다룰 것이다.



summary

핵심 정리



- 제어문이란 프로그램의 순차적 흐름이 아닌 조건에 맞는 실행과 반복을 하는 명령문이다.
- 조건문이란 프로그래머가 명시한 특정 조건을 만족할 때에만 특정 블록이 실행되도록 하는 문법이다.
- 조건문의 키워드인 **if**, **elif**의 다음에는 조건식이 나타나고 바로 이어서 실행문 블록이 시작되기 전에 콜론(:)을 표시한다. 그리고 실행문 블록은 조건문의 키워드보다 들여쓰기를 해서 작성한다.
- 조건식은 부울형인 참(**True**)과 거짓(**False**)의 값을 가질 수 있다.
- 블록은 같은 실행 흐름에서 순서대로 실행되는 코드의 집합을 의미한다.
- 조건문에서 실행문 블록은 반드시 들여쓰기를 해주어야 한다.
- **if-elif-else** 문에서 **if** 문의 조건식을 제일 먼저 조사하고, **if** 문의 조건식이 참이 아닐 때, **elif** 조건식을 검사하고, **elif** 문의 조건식도 참이 아닐 때에는 마지막으로 **else** 문의 블록을 실행한다.

따라하며 배우는

파이썬과 데이터 과학



Questions?