05 제어 구조와 데이터 구조

✚ **학습 목표** •제어 구조를 복합적으로 활용한 프로그램을 작성할 수 있다.
•다차원 데이터 구조를 활용하여 프로그램을 작성할 수 있다.

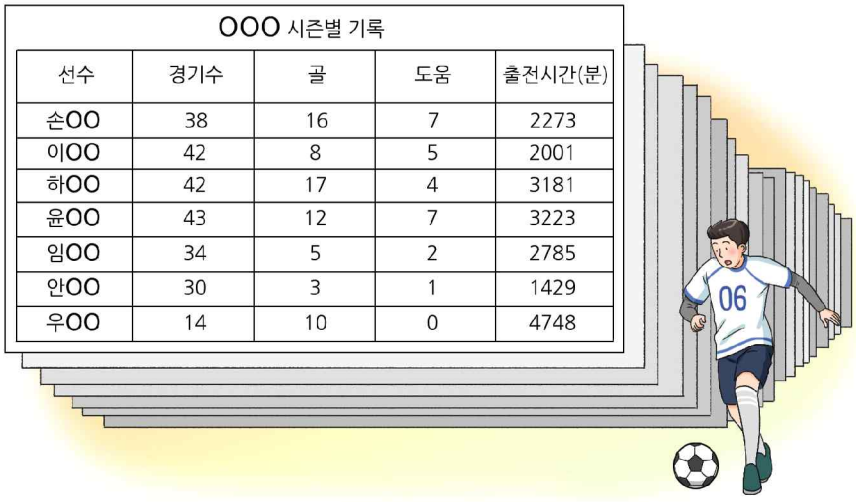
✚ **학습 요소** •제어 구조, 데이터 구조, 순차 구조, 선택 구조, 반복 구조, 리스트

생각 깨우기 **축구 선수 기록의 효율적 관리에 대해 생각해 보자.**

축구 선수의 기록은 경기수, 골, 도움 등을 속성으로 방대한 양이 저장되고 관리된다.

○○○ 시즌별 기록

선수	경기수	골	도움	출전시간(분)
손OO	38	16	7	2273
이OO	42	8	5	2001
하OO	42	17	4	3181
윤OO	43	12	7	3223
임OO	34	5	2	2785
안OO	30	3	1	1429
우OO	14	10	0	4748



Q **프로그램을 활용하여 효율적으로 데이터를 관리하기 위한 방법은 무엇이 있을까?**

1 제어 구조를 활용한 프로그래밍

01 제어 구조의 이해

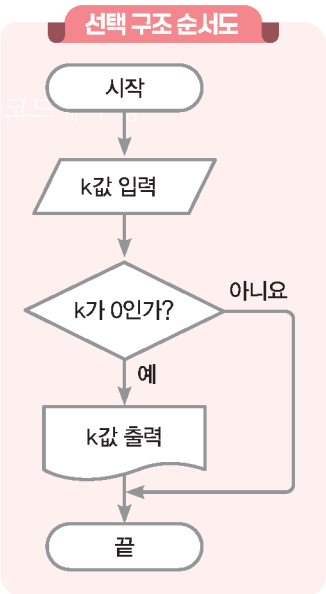
순차 구조는 주어진 명령을 순서대로 실행한다. 그러나 조건에 따라 다른 명령을 실행하거나 특정 명령을 반복 실행해야 하는 경우 제어 구조를 사용하는데, 제어 구조는 프로그램의 처리 과정을 제어하기 위한 구조를 의미한다.

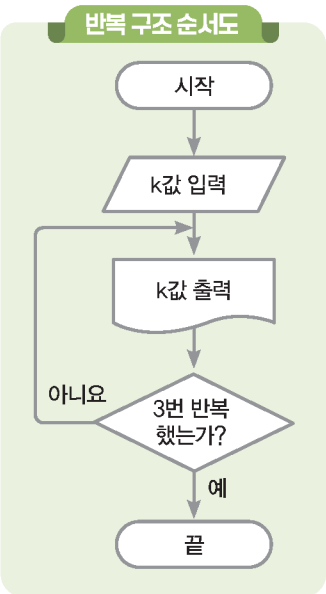
선택 구조는 주어진 조건에 따라 프로그램의 흐름을 달리하고자 할 때 사용하는 구조다.

예를 들어 k값을 입력받았을 때 k가 0이면 k값을 출력하는 프로그램은 왼쪽 그림과 같다. 먼저 k를 입력받고, k가 0이라면 k값을 출력한 다음 프로그램을 종료하고, k가 0이 아니라면 바로 프로그램을 종료한다.

반면 반복 구조는 같은 명령을 반복하여 실행할 때 사용하는 구조다. 프로그램 코드에서 동일 부분의 반복 횟수가 많아질수록 코드를 중복하여 사용하는 것은 효율적이지 않다. 따라서 동일한 명령이 반복될 때 반복문을 활용하여 이를 간결하게 표현하는 것이 효율적이다.

예를 들어 k값을 입력받아 k값을 3번 반복하여 출력하는 프로그램은 오른쪽 그림과 같다. 먼저 k값을 입력받고 k값을 출력한다. 본 과정을 3번 반복했는지 확인하고, 3번 반복하지 않았다면 k값을 반복하여 3번 출력한 후 프로그램을 종료한다.





02 선택 구조의 활용

프로그램의 흐름을 제어하는 선택 구조는 if~else문, if~elif~···~else문이 있다.

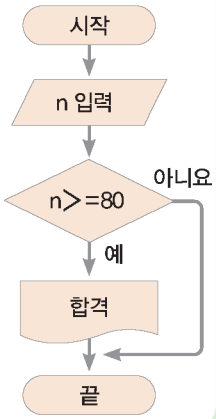
1 if문

조건식이 참이면 문장 1을 실행한다.

if문

```
if 조건식:
  문장 1
```

〈실습 1〉의 순서도



실습 1

한 개의 정수를 입력받아 80 이상이면 '합격'을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 n = int(input('입력: '))          # n은 점수를 의미하는 변수
2 if n > = 80:                      # n이 80 이상이면 합격 출력
3   print('합격')
```

실행 결과

```
입력: 85
출력: 합격
```

설명 입력받은 정수 n이 80보다 크거나 같다면 합격을 출력한다. 예를 들어 85가 입력되면 합격이 출력된다.

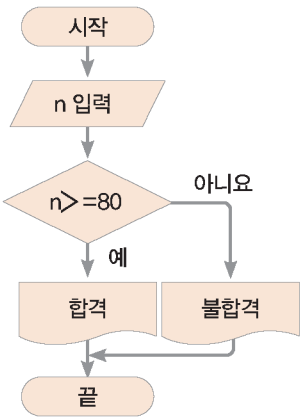
2 if~else문

조건식이 참이면 문장 1을 실행하고 거짓이면 문장 2를 실행한다.

if~else문

```
if 조건식:
  문장 1
else:
  문장 2
```

〈실습 2〉의 순서도



실습 2

한 개의 정수를 입력받아 80 이상이면 '합격', 80 미만이면 '불합격'을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 n = int(input('점수'))            # n은 점수를 의미하는 변수
2 if n > = 80:                      # n이 80 이상이면 합격 출력
3   print('합격')
4 else:                             # 조건식이 거짓이면 불합격 출력
5   print('불합격')
```

실행 결과

```
입력: 75
출력: 불합격
```

실행 결과

```
입력: 85
출력: 합격
```

설명 입력받은 정수 n이 80보다 크거나 같다면 '합격'을 출력하고, 그렇지 않다면 '불합격'을 출력한다. 예를 들어 75가 입력되면 '불합격', 85가 입력되면 '합격'을 출력한다.

❸ if~elif~…~else문

조건식 1이 참이면 문장 1을 실행하고, 조건식 2가 참이면 문장 2를 실행한다. 모든 조건식이 거짓이면 문장 3을 실행한다. if와 else 사이의 elif는 필요한 만큼 추가할 수 있다.

if~elif~…~else문

```
if 조건식 1:
  문장 1
elif 조건식 2:
  문장 2
⋮
else:
  문장 3
```

else는 반드시 있을 필요가 없으며 만일 else가 없고, 참인 조건이 없다면 아무것도 실행하지 않고 다음 문장을 실행한다.

실습 3

한 개의 정수를 입력받아 80 이상이면 '합격', 80 미만 60 이상이면 '재시험', 60 미만이면 '불합격'을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 n = int(input('입력: '))     # n은 점수를 의미하는 변수
2 if n >= 80:                  # n이 80 이상이면 합격 출력
3   print('합격')
4 elif(n < 80 and n >= 60):    # n이 60 이상이면 재시험 출력, elif n >= 60으로 작성 가능
5   print('재시험')
6 else:                        # 조건식이 모두 거짓이면 불합격 출력
7   print('불합격')
```

파이썬에서 들여쓰기는 실행 범위를 결정하므로 반드시 규칙에 맞게 지켜야 한다. 들여쓰기를 하지 않는 경우 오류가 발생한다. 일반적으로 if, while, for문에서 :(콜론) 다음의 문장은 반드시 탭(Tap) 키로 들여쓰기해야 한다.

실행 결과

```
입력: 55
불합격
```

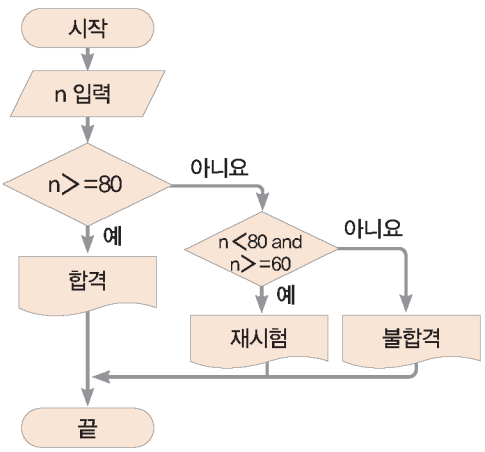
실행 결과

```
입력: 75
재시험
```

실행 결과

```
입력: 85
합격
```

〈실습 3〉의 순서도

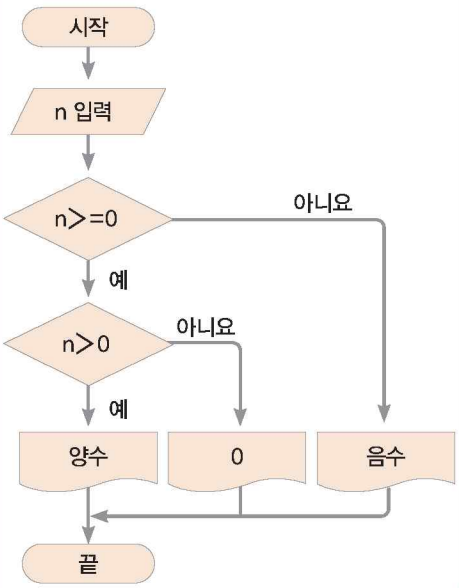


설명 입력받은 정수 n이 80보다 크거나 같다면 '합격', 80 미만이고 60 이상이면 '재시험', 두 조건 모두 거짓이면 '불합격'을 출력한다. 예를 들어 75가 입력되면 80 미만이면서 60 이상이므로 '재시험'을 출력한다.

4 중첩 선택 구조

복잡한 문제를 해결할 때에는 다양한 선택 구조를 중첩하여 문제를 해결해야 한다. 중첩 선택 구조란 선택 구조 안에 다른 선택 구조가 중첩되어 있는 것으로 이를 활용하면 여러 가지 상황에 따라 실행할 수 있는 프로그램을 만들 수 있다.

〈실습 4〉의 순서도



실습 4

숫자를 입력받아 양수, 음수, 0을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 n = int(input('입력: ')) # 입력받은 숫자를 저장하는 변수
2 if n >= 0:
3   if n > 0:
4     print('양수')
5   else:
6     print('0')
7 else:
8   print('음수')
```

실행 결과

```
입력: -5
음수
```

실행 결과

```
입력: 5
양수
```

실행 결과

```
입력: 0
0
```

설명 입력받은 정수 n이 0 이상이면서 0보다 크다면 양수, 그렇지 않다면 0을 출력한다. 입력받은 정수 n이 0 이상이 아니라면 음수를 출력한다.

알고 가기 비교 연산자와 논리 연산자

비교 연산자와 논리 연산자는 선택 구조, 반복 구조의 조건에 유용하게 활용된다.

비교 연산자		논리 연산자	
a>b	a가 b보다 큰가?	a and b	a와 b가 동시에 참일 때만 전체 값이 참이 된다.
a<b	a가 b보다 작은가?		
a>=b	a가 b 이상인가?	a or b	a와 b 중 하나 이상이 참이면 전체 값이 참이 된다.
a<=b	a가 b 이하인가?		
a==b	a와 b가 같은가?	not a	a가 참이면 거짓, a가 거짓이면 참이 된다.
a!=b	a와 b가 다른가?		

해 보기 1 이중 암호 해제하기

다음 **문제 상황**을 읽고, 이것을 해결하기 위한 프로그램을 완성하고 실행해 보자.

문제 상황 나의 휴대폰 비밀번호를 이중으로 설정 및 해제하고자 한다. 첫 번째 비밀번호는 4자리의 숫자(1004)로, 두 번째 비밀번호는 4자리 숫자와 문자 1 글자로 설정하고자 할 때 이를 확인하기 위한 프로그램을 완성해 보자.

방법 1

프로그램

```
pwn = 1004                                  # pwn: 휴대폰 숫자 비밀번호 4자리
pwa = 'a'                                   # pwa: 휴대폰 문자 비밀번호 1자리
n = int(input('숫자 비밀번호 입력: '))       # n: 사용자가 입력한 숫자 비밀번호
a = input('문자 비밀번호 입력: ')            # a: 사용자가 입력한 문자 비밀번호
if [          ]:
  print('잠금이 해제되었습니다.')
elif [          ]:
  print('문자 비밀번호가 잘못되었습니다.')
elif [          ]:
  print('숫자 비밀번호가 잘못되었습니다.')
else:
  print('숫자, 문자 비밀번호가 잘못되었습니다.')
```

방법 2

프로그램

```
pwn = 1004                                  # pwn: 휴대폰 숫자 비밀번호 4자리
pwa = 'a'                                   # pwa: 휴대폰 문자 비밀번호 1자리
n = int(input('숫자 비밀번호 입력: '))       # n: 사용자가 입력한 숫자 비밀번호
a = input('문자 비밀번호 입력: ')            # a: 사용자가 입력한 문자 비밀번호
if [          ]:
  if [          ]:
    print('잠금이 해제되었습니다.')
  else:
    print('문자 비밀번호가 잘못되었습니다.')
else:
  if [          ]:
    print('숫자 비밀번호가 잘못되었습니다.')
  else:
    print('숫자, 문자 비밀번호가 잘못되었습니다.')
```

03 반복 구조의 활용

반복 구조는 조건에 따라 특정 명령을 반복 실행하는 구조로 for문과 while문이 있다.

1 for문

for문은 리스트, range 등을 활용하여 반복문을 실행한다.

for문

```
for 변수명 in 리스트(또는 문자열):
    문장
```

설명 리스트(또는 문자열)의 첫 번째 요소부터 마지막 요소까지 차례대로 변수에 대입하여 문장을 반복하여 실행한다.

리스트 속 자료를 출력하는 프로그램을 실습해 보자.

실습 5

숫자를 저장한 리스트를 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 for i in [1, 5, 3, 10]:
2   print(i)
```

실행 결과

```
1
5
3
10
```

설명 반복문 for에서 리스트 [1, 5, 3, 10]에 있는 숫자를 1개씩 출력한다.

실습 6

문자열을 저장한 리스트를 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 animal = ['강아지', '고양이', '코끼리', '기린'] # animal 리스트 생성
2 for i in animal:
3   print(i)
```

실행 결과

```
강아지
고양이
코끼리
기린
```

설명 animal 리스트를 생성하고, animal 리스트에 들어 있는 값을 1개씩 출력한다.

range 함수는 규칙이 반복적인 숫자를 사용할 때 활용하는 함수로, 반복문에서 유용하게 활용된다. range(시작 값, 종료 값, 증감 값)의 형태로 활용하며, 시작 값부터 종료 값 직전까지 증감 값만큼 변화함을 의미한다.

range를 활용한 반복문을 실습해 보자.

실습 7

range(5)를 활용한 반복문 프로그램을 작성하고 실행해 보자.

프로그램

```
1 for i in range(3):
2   print(i)
```

실행 결과

```
0
1
2
```

range 함수에서 시작 값, 종료 값은 입력하지 않을 수 있다.

설명 range(3)을 통해 0, 1, 2의 수를 출력한다.

실습 8

range(1, 3)를 활용한 반복문 프로그램을 작성하고 실행해 보자.

프로그램

```
1 for i in range(1, 3):
2   print(i)
```

실행 결과

```
1
2
```

설명 range(1, 3)은 1부터 2까지의 수를 출력하는 것을 의미한다. 따라서 1, 2를 출력한다.

실습 9

range(5, 0, -2)를 활용한 반복문 프로그램을 작성하고 실행해 보자.

프로그램

```
1 for i in range(5, 0, -2):
2   print(i)
```

실행 결과

```
5
3
1
```

파이썬에서는 산술 계산을 편리하게 할 수 있는 다양한 함수를 사용할 수 있다. 1부터 100까지 수의 합을 구하는 프로그램을 다음과 같이 구현할 수 있다.

프로그램

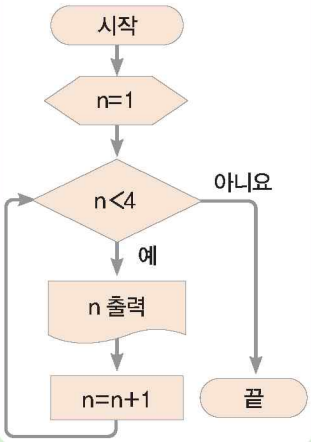
```
s = sum(range(1,101,1))
print(s)
```

설명 range(5, 0, -2)를 통해 5부터 시작하여 1까지 2씩 작아지는 값을 출력한다.

2 while문

while문은 조건식을 사용한다. 조건식이 참이면 명령문을 실행하고 반복문을 반복하며, 조건식이 거짓이면 반복을 종료한다.

〈실습 10〉의 순서도



while문

```
while 조건식:
      문장
```

실습 10

1부터 3까지 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 n = 1
2 while n < 4:
3   print(n)
4   n = n + 1
```

실행 결과

```
1
2
3
```

설명 n이 4보다 작다면 n을 출력하고 n에 1을 더하는 과정을 반복한다. n에 처음에 1이 저장되었으므로 4보다 작은 1, 2, 3이 출력된다.

해 보기 2 숫자의 합 계산하기

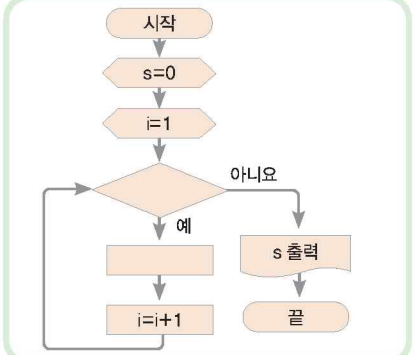
1 1부터 100까지 수의 합을 구하는 코드와 순서도를 완성해 보자.

프로그램 코드(for 활용)

```
1 s = 0      # 합계를 저장하는 변수
2 for i in range [          ]:
3   s = [          ]
4 print(s)
```

프로그램 코드(while 활용)

```
1 s = 0      # 합계를 저장하는 변수
2 i = 1
3 while [          ]:
4   s = [          ]
5   i = i + 1
6 print(s)
```



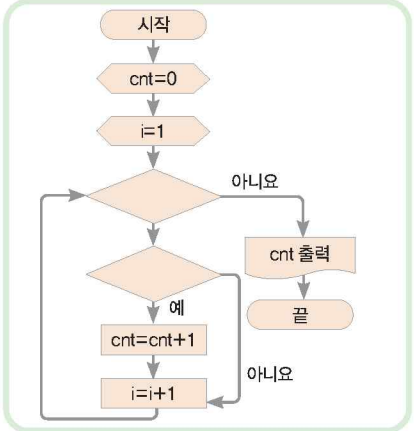
2 1부터 70 사이의 수 중 7의 배수의 개수를 구하는 코드를 작성해 보자.

프로그램 코드(for 활용)

```
1 cnt = 0
2 for i in range [          ]:
3   if [          ]:
4     cnt = cnt + 1
5 print(cnt)
```

프로그램 코드(while 활용)

```
1 cnt = 0
2 i = 1
3 while [          ]
4   if [          ]
5     cnt = cnt + 1
6   i = i + 1
7 print(cnt)
```



❸ 중첩 반복 구조

중첩 제어 구조

반복문 속 반복문, 조건문 속 반복문 등 여러 개의 제어 구조를 중첩하여 사용하는 것을 말한다. 상황에 따라 여러 제어 구조를 중첩하여 사용할 수 있다.

실습 11

중첩 반복문으로 숫자를 출력하는 프로그램을 작성하고 실행해 보자.

프로그램 ❶

```
1  # for 활용
2  for i in range(1, 4, 1):
3    for j in range(1, i+1, 1):
4      print(j, end = ' ')
5    print('')    # 1줄 출력 후 줄바꿈
```

실행 결과

```
1
1 2
1 2 3
```

설명 프로그램 ❶의 경우 첫 번째 for문에서는 range(1, 4, 1)을 통해 1부터 3까지의 수가 반복된다. 두 번째 for문에서는 앞에서 설정된 i값(1~3)에 따라 j값이 (1부터 i까지) 반복되어 출력되고 줄을 바꾼다. 따라서 i가 1일 때 j값 1이 출력된 후 줄이 바뀌고, i가 2일 때 j값 1과 2가 출력된 후 줄이 바뀐다. 마찬가지로 i가 3일 때 j값 1 2 3을 출력하고 줄을 바꾼다.

print를 실행하면 자동으로 줄바꿈이 된다. print로 출력되는 결과를 한 줄로 이어서 출력하기 위해서는 end = ' ' 옵션을 추가한다.

프로그램 ❷

```
1  # while 활용
2  i = 1
3  j = 1
4  while i <= 3:
5    while j <= i:
6      print(j, end = ' ')
7      j = j + 1
8    print('')
9    i = i + 1
10   j = 1
```

실행 결과

```
1
1 2
1 2 3
```

설명 프로그램 ❶과 같은 결과가 출력되는 프로그램이다. i는 1부터 3까지 반복되며, j는 1부터 i까지 반복되며 값을 출력한다.

해 보기 3 별 출력하기

1 중첩 반복문을 활용하여 네모 모양을 출력해 보자.

출력

```
***
***
***
```

2 중첩 반복문을 활용하여 삼각형 모양을 출력해 보자.

출력

```
*
**
***
```

2 | 데이터 구조를 활용한 프로그래밍

01 데이터 구조의 활용

데이터 구조는 컴퓨터에서 효율적으로 자료를 처리하고 관리하기 위한 구조로, 리스트가 주로 활용된다. 리스트와 반복문을 사용하면 수많은 변수를 필요로 할 때 효율적인 프로그래밍을 할 수 있다. 리스트는 1차원 리스트와 2차원 리스트 등이 있으며, 필요에 따라 원하는 형태로 선언하여 사용할 수 있다. 예를 들어 한 학생의 이름, 성별, 학년, 정보 점수, 인공지능 점수 등을 표현하려면 5개의 변수가 있어야 하지만 리스트를 활용하면 단 1개의 리스트에 저장할 수 있어 효율적이다.

5개의 자료를 1차원 리스트에 저장하고, 출력하는 방법은 다음과 같다.

❶ 1차원 리스트 생성하기

리스트 이름 = [값1, 값2, 값3, …]

❷ 1차원 리스트의 구조

s[0]	s[1]	s[2]	s[3]	s[4]
김OO	남	1학년	85	90

리스트의 값은 순서가 있으며 그 순서를 인덱스로 표현한다. 인덱스 번호는 0부터 시작하며, 인덱스 번호를 사용하여 특정 리스트의 값에 접근할 수 있다.

❸ 1차원 리스트 출력하기

리스트의 값을 하나하나 출력하기보다는 반복문을 사용하여 출력하는 것이 효율적이다. 모든 리스트의 값을 출력할 때에는 각 리스트의 인덱스를 처음부터 끝까지 변환하며 출력한다.

실습 12

1차원 리스트를 생성하고 리스트의 모든 값, 특정 값을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
1 s = ['김OO','남','1학년', 85, 90]
2 for i in range(0, 5):                  # 1차원 리스트의 모든 값 출력하기
3   print(s[i], end=' ')
4   print(' ')
5 print(s[2])                            # 1차원 리스트의 특정 값 출력하기
```

print에서 end=' '은 print에서 값을 출력한 후 한 칸 띄어 출력하는 코드다. 만일 end=' '를 작성하지 않는다면 값을 출력한 후 자동으로 줄을 변경한다.

실행 결과

```
김OO 남 1학년 85 90
1학년
```

설명 김OO, 남, 1학년, 85, 90의 값을 저장한 1차원 리스트 s를 생성한다. 이어서 range(0, 5)에 따라 0부터 4까지의 값이 i 값으로 반복되며 s[i] 의 값을 순서대로 출력한다. 이어서 s[2]에 저장되어 있는 1학년을 출력한다.

02 다차원 데이터 구조의 활용

다차원 데이터 구조는 여러 차원으로 데이터를 저장하여 효과적으로 데이터를 표현하고 관리하는 데 활용된다. 만일 여러 속성을 가진 데이터를 1차원으로 표현한다면 다양한 속성을 한눈에 알아보기 어렵지만 2차원으로 표현한다면 다양한 속성을 알아보기 쉽다. 따라서 2명 이상 학생의 자료를 저장할 때에는 2차원 리스트를 사용하는 것이 효율적이다. 2차원 리스트를 활용하여 학생 5명의 자료(이름, 성별, 학년, 정보 점수, 인공지능 점수)를 저장하는 방법은 다음과 같다.

- 리스트의 최대 차원은 제한이 없다. 하지만 차원이 높아질수록 메모리 사용량과 연산 속도가 증가할 수 있음을 유의해야 한다.
- 다차원 데이터 구조는 이미지 및 비디오 저장을 위한 데이터, 기계학습 모델에 활용되는 데이터, 여러 속성을 가진 데이터 등을 저장하고 처리하는 데 활용된다.

1 2차원 리스트 생성하기

리스트 이름 = [[값1, 값2, 값3], [값4, 값5, 값6], …

2 2차원 리스트의 구조

	s[ ][0]	s[ ][1]	s[ ][2]	s[ ][3]	s[ ][4]
s[0][ ]	김OO	남	1학년	85	90
s[1][ ]	이OO	여	2학년	75	95
s[2][ ]	박OO	여	1학년	90	70
s[3][ ]	정OO	여	1학년	70	80
s[4][ ]	한OO	남	2학년	95	75

3 2차원 리스트 출력하기

2차원 리스트의 값을 출력할 때에는 중첩 반복 구조를 사용하면 효율적이다. 리스트 이름[행][열]을 고려하여 리스트의 값을 출력할 수 있다.

실습 13

2차원 리스트를 생성하고 리스트의 모든 값, 특정 값을 출력하는 프로그램을 작성하고 실행해 보자.

프로그램

```
s = [['김OO', '남', '1학년', 85, 90],\
     ['이OO', '여', '2학년', 75, 100],\
     ['박OO', '여', '1학년', 90, 70],\
     ['정OO', '여', '1학년', 65, 80],\
     ['한OO', '남', '2학년', 95, 75]]

# 2차원 리스트의 모든 값 출력하기
for i in range(0, 5):          # 1행~5행까지 출력
  for j in range(0, 5):        # 1열~5열까지 출력
    print(s[i][j], end=' ')# 각 리스트의 값 출력
  print(' ')                   # 줄 바꿈

# 2차원 리스트의 특정 값 출력하기
print(s[3][4])
```

실행 결과

```
김OO 남 1학년 85 90
이OO 여 2학년 75 100
박OO 여 1학년 90 70
정OO 여 1학년 65 80
한OO 남 2학년 95 75
```

실행 결과

```
80
```

- print(s) 명령어로 리스트의 내용을 한 번에 출력할 수 있다.

설명 2차원 리스트의 값들을 s에 저장한다. 리스트는 총 5개의 행과 열로 이루어져 있으므로 반복문을 활용하여 인덱스 0~4까지 행과 열을 변환시켜 가며 값을 출력하도록 한다. 이어서 s[3][4]에 저장되어 있는 값 80을 출력한다.

해 보기 4 급식 메뉴 선호 설문 조사 결과 처리하기

다음은 급식 메뉴 선호 설문 조사 결과다. 주어진 문제를 다차원 데이터 구조와 제어 구조를 활용하여 해결해 보자.

제육볶음	생선구이	스파게티	돈가스
42	23	77	58

1 급식 메뉴 선호 설문 조사 결과를 저장한 2차원 리스트를 생성해 보자.

프로그램

```
```

2 설문 조사에 참가한 학생 수의 합을 구해 보자.

프로그램

```
1 sum = 0                          # 학생 수 합계 변수 sum
2 for i in range(4):
3     [                          ]
4 print(sum)
```

3 설문 조사 결과 투표 수가 50 이상인 메뉴를 출력해 보자.

프로그램

```
1 for i in range(4):
2     if [                      ] :
3         print( [                      ] )
```

소단원 1분 요약

❶ 제어 구조는 프로그램의 처리 과정을 제어하기 위한 구조로, 순차 구조, 선택 구조, 반복 구조가 있다.

❷ 선택 구조에는 if문, if~else문, if~elif~...~else문이 있고, 반복 구조에는 for문, while문이 있다.

❸ 데이터 구조는 여러 데이터를 관리하기 쉽도록 같은 이름으로 생성하여 관리하는 구조로, 리스트가 주로 활용된다.

소단원 자기 평가

평가 항목	평가 기준		
	잘함	보통	노력
1. [지식이해] 제어 구조의 응용 방법을 설명할 수 있다.			
2. [지식 이해] 다차원 데이터 구조의 적용 방법을 설명할 수 있다.			
3. [과정기능] 복잡한 문제를 해결하기 위해 다양한 제어 구조와 다차원 데이터 구조를 복합적으로 활용할 수 있다.			

정보 탐구 활동

방석 주문 제작 프로그램 작성하기

A 방석 회사는 정사각형, 원형, 정삼각형 모양의 주문형 방석을 제작하고자 한다. 주문 받은 방석 목록이 다음과 같을 때 방석의 넓이를 계산하는 프로그램을 작성해 보자.

원형	정사각형	정사각형	원형	정삼각형
5	14	8	10	12

※ 목록의 속성은 형태와 반지름(혹은 변의 길이)이다.

1 order라는 이름의 2차원 리스트를 만들고, 주문 받은 방석 목록을 저장해 보자.

프로그램

```
```

2 각 방석의 넓이를 계산하여, 출력하는 프로그램의 빈칸을 완성해 보자.

프로그램

```
1 import math                                  # 제곱근을 구하는 sqrt 함수를 사용하기 위한 라이브러리 추가
2 for i [          ]
3   if order[0][i] == '원형':
4     print(order[1][i] * order[1][i] * 3.14)
5   else:
6     if order[0][i] == '정사각형':
7       print(order[1][i] * order[1][i])
8     else:
9       print(order[1][i] * order[1][i] * math.sqrt(3) / 4) # math.sqrt(3)은 √3을 의미
```

탐구 활동 자기 평가	평가 항목	평가 기준		
		잘함	보통	노력
	1. 다차원 데이터 구조를 활용하여 프로그램을 작성할 수 있는가?			
	2. 제어 구조를 복합적으로 활용하여 프로그램을 작성할 수 있는가?			

• 잘함 (내용을 이해하고 설명함) • 보통 (내용을 이해함) • 노력 (내용을 부분적으로 이해함)