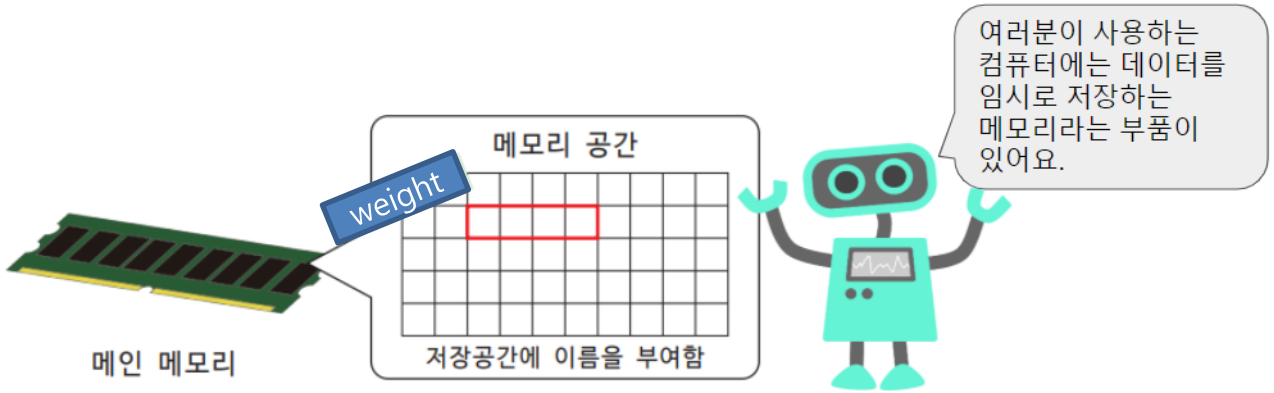
데이터분석 프로그래밍
변수

임현기

변수

- 데이터분석을 위해 일단 데이터를 저장해야 함

- 파이썬에서는 변수에 데이터를 저장
  - 정수, 실수, 문자열 등의 자료 값을 저장 가능



변수

- 저장된 것을 꺼내려면 변수의 이름을 적음
- 상자에 저장된 값은 수시로 바뀔 수 있음

변수

- 예: 자신의 몸무게를 weight라는 이름의 변수에 저장

```
>>> weight = 78.2
```

- 파이썬 내부에 weight라는 이름의 변수가 생성되고 여기에 78.2가 저장
  - 위의 코드에 있는 '=' 기호는 '같다'는 등호의 의미가 아님
  - '오른쪽의 값을 왼쪽의 weight라는 이름의 변수에 저장하라'는 의미
  - 이 연산자 '=' 을 할당연산자 혹은 대입연산자라고 함

변수

- 변수의 값을 출력

```
>>> weight
78.2
```

- 파이썬의 대화창 콘솔에서는 변수의 이름만 써 주어도 변수의 값이 출력 가능
- 스크립트 파일에서는 반드시 print(weight) 함수를 사용하여야 변수의 값을 출력 가능

```
weight = 78.2
print(weight)
```

```
78.2
```

변수 값 변경

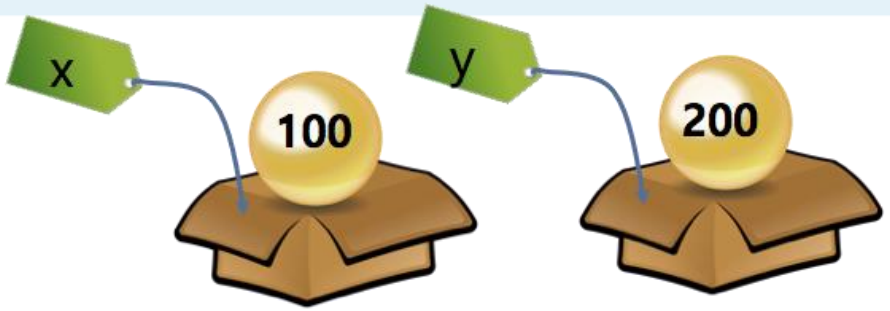
- 변수에 있는 값은 언제든지 바꿀 수 있음
- 예: 체중 변화

```
>>> weight = 78.2     # weight 변수의 최초 값
>>> weight = 76.9     # 변수가 가지고 있는 값은 변경할 수 있다
>>> weight
76.9
```

변수 값 변경

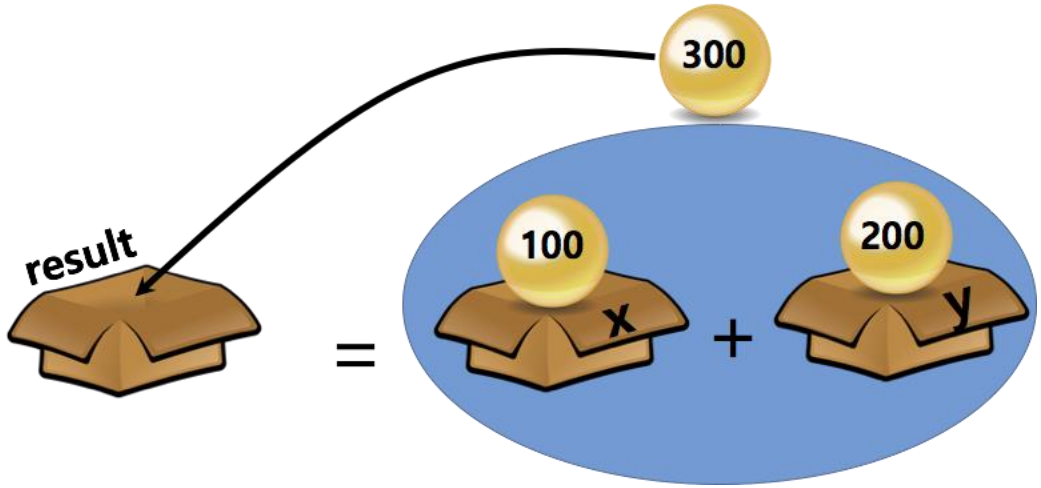
- 2개의 변수 x와 y를 생성하고 여기에 100과 200을 저장

```
>>> x = 100
>>> y = 200
>>> x, y
(100, 200)
```



변수 값 변경

- x, y = 100, 200과 같이 한 줄에 여러 개의 변수를 선언하고 이 변수에 값을 동시에 할당할 수 있음
- x + y 연산하여 그 결과 값을 변수 result에 저장



변수 이름

- 변수의 이름은 몇가지 규칙을 지켜야 함
- 변수의 이름은 식별 역할을 함

변수 이름

- 변수 이름 규칙
  - 식별자는 문자와 숫자, 밑줄 문자(_)로 이루어지며 밑줄 문자 이외의 특수 문자를 사용할 수 없음
  - 식별자의 첫 글자는 숫자로 시작할 수 없음
  - 중간에 공백을 가질 수 없음
  - 대문자와 소문자는 구별 (변수 index와 INDEX은 서로 다른 변수)
  - 파이썬의 예약어(키워드)는 식별자로 사용할 수 없음

변수 이름

- 예약어

False	class	return	is	finally	None
if	for	lambda	continue	True	def
from	while	nonlocal	and	del	global
not	with	as	elif	try	or
yield	assert	else	import	pass	break
except	in	raise			

변수 이름

- 변수 이름은 변수의 역할을 가장 잘 설명하는 이름으로 지어야 함
  - 프로그램 가독성을 높임

체중과 키를 저장하는 나쁜 변수 이름	체중과 키를 저장하는 좋은 변수 이름
`>>> x1 = 78.2` `>>> x2 = 180.0`	`>>> weight = 78.2` `>>> height = 180.0`

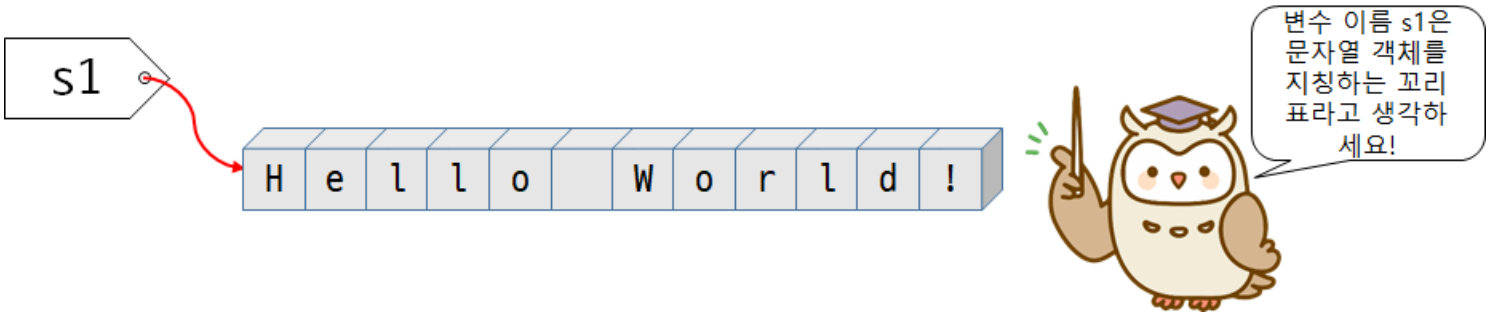
문자열 변수

- 문자열은 큰따옴표("...")나 작은따옴표('...')를 이용하여 만들 수 있음
- "Hello World!"도 문자열, 'Hello World!'도 같은 문자열

```
>>> s1 = 'Hello World!'
>>> s1
'Hello World!'
```

문자열 변수

- 파이썬에서는 문자열 데이터를 '객체'라는 용어로 부르며, 변수 s1은 이 객체를 지칭하는 꼬리표로 봄



  - 편리하게 호출하여 사용할 수 있는 미리 정의된 기능을 함수라는 형태로 제공
  - s1 문자열의 길이는 문자열에 len() 함수를 적용하여 알 수 있음

```
>>> len(s1)    # s1 문자열의 길이를 알려준다
12
```

문자열 변수

- 문자열 덧셈, +연산

```
>>> s1 = "Hello"
>>> s2 = "World!"
>>> s1 + s2       # s1 문자열과 s2 문자열을 결합한 결과를 반환
'HelloWorld!'
>>> n1 = '100'
>>> n2 = '200'
>>> n1 + n2       # 문자열에 대한 덧셈 연산자는 문자열 이어 붙이기만 수행
'100200'
```

자신의 키와 몸무게를 변수에 저장하고 이들 변수들을 이용하여 BMI를 계산해보자. BMI는 신체 질량 지수로서 신장과 체중을 이용하여 계산할 수도 있다. BMI는 사람의 체지방량과 상관관계가 크다고 증명된 바 있다. BMI 지수는 킬로그램 단위로 측정한 체중을 w, 미터 단위로 측정한 키가 h라고 할 때 다음과 같이 구할 수 있다.

$$BMI = \frac{w}{h^2}$$

원하는 결과

체중(kg), 키(m)를 입력하면 자동으로 BMI 값이 계산되어 변수 bmi에 저장되고, 그 값을 다음과 같이 출력하도록 해보자.

```
>>> bmi
24.1358024691358
```

```
>>> height = 1.80
>>> weight = 78.2
>>> bmi = weight / height**2
>>> bmi
24.1358024691358
```

변수 이용의 장점

- 기존 스크립트

```
>>> height = 1.80
>>> weight = 78.2
>>> bmi = weight / height**2
>>> bmi
24.1358024691358
```

- weight 값 변경 후 스크립트 실행

```
>>> height = 1.80
>>> weight = 72.8
>>> bmi = weight / height**2
>>> bmi
22.469135802469133
```

변수 이용의 장점

- 가독성을 높임
  - 아래 예제는 의미 파악이 어려움

```
>>> print(3.141592 * 10 * 20)
628.3184
```

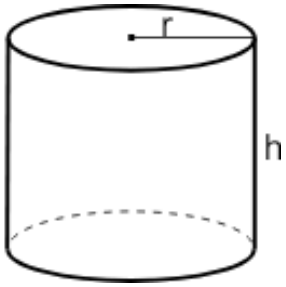
```
>>> print(3.141592 * (10 ** 2) * 20)
6283.184
```

변수 이용의 장점

- 가독성을 높임

자신의 코드를 나중에 보는 경우가 많으니, 체계적으로 코딩하고, 변수의 이름도 이해하기 쉽게 하세요.

$V = h \cdot \pi \cdot r^2$




```
>>> pi = 3.141592
>>> radius = 10
>>> height = 20
>>> volume_of_cylinder = pi * (radius ** 2) * height
>>> print(volume_of_cylinder)
6283.184
```

다음은 연 3%의 복리 이자를 주는 금융 상품에 1,000만원을 저축하였을 때 5년 후에 얻게 되는 돈을 계산하는 수식이다.

$$10,000,000 \times (1 + 0.03)^5$$

이것을 파이썬 문법에 맞춰 표현하면 다음과 같다.

```
>>> 10000000 * (1.0 + 0.03) ** 5
11592740.743
```

이 수식에 나타나는 값들을 변수로 변경해서 다시 작성해 보자. 그러면 원금과 이율을 바꾸어 가며 복리 계산을 할 수 있을 것이다.

원하는 결과

```
원금:    10000000
이율:    0.03
기간:    5
수령금액:   11592740.743
```

```python
principal = 10000000
years = 5
interest_rate = 0.03
money = principal * (1.0 + interest_rate) ** years

print('원금:   ', principal)
print('이율:   ', interest_rate)
print('기간:   ', years)
print('수령금액:  ', money)
```

자료형

- 파이썬은 기본 자료형 4가지를 제공

자료형	예
정수(int)	…, -3, -2, -1, 0, 1, 2, 3, ...
실수(float)	3.14, 4.28, 0.01, 123.432
문자열(str)	'Hello World', '123', "Hi"
부울형(bool)	True, False

자료형

- 변수에 어떤 종류의 데이터도 저장 가능
- 데이터 변경도 가능

```
>>> x = 10     # 정수 10을 변수 x가 저장하게 됨
>>> x
10
>>> x = 3.14   # 변수 x가 저장하는 값이 3.14로 바뀐다
>>> x
3.14
```

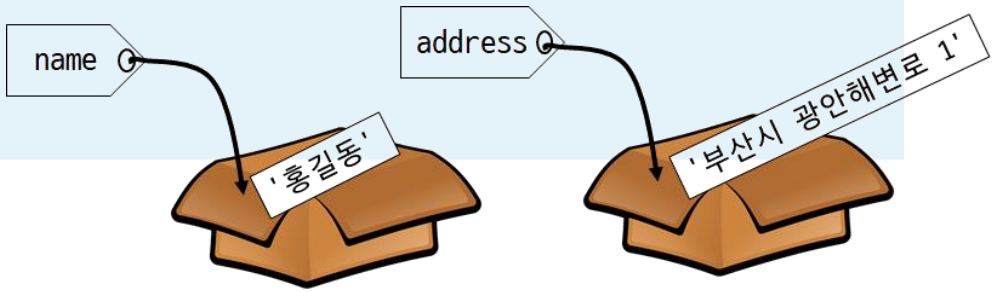
자료형

- 자료형을 알기 위해 type() 함수 사용

```
>>> weight = 78.2
>>> type(weight)
<class 'float'>
```

```
>>> salary = 250
>>> type(salary)
<class 'int'>
```

```
>>> name = '홍길동'
>>> address = '부산시 광안해변로 1'
>>> type(name)
<class 'str'>
>>> type(address)
<class 'str'>
```




```
>>> b = True
>>> type(b)
<class 'bool'>
```

자료형

- 자료형에 따라 연산의 의미가 달라짐

```
>>> x = 10
>>> y = 10
>>> x + y
20
>>> x = 'good'
>>> y = 'morning!'
>>> x + y
'goodmorning!'
```

x, y는 각각 10, 10을 참조하는 정수형

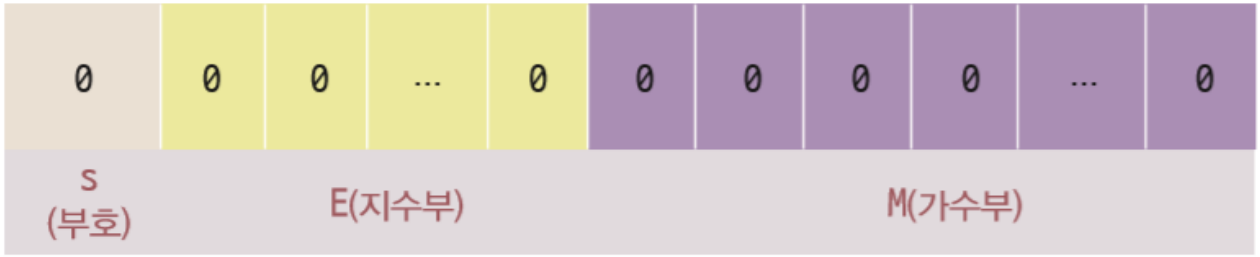
x, y는 각각 'good', 'morning'을 참조하는 문자열형
(자료형이 변경되어 +연산이 하는 일이 달라짐)

```
>>> '23' + '56'
'2356'
>>> 23 + 56
79
```

수치 표현의 한계

- 1/3을 소수로 표현하면?
  - 컴퓨터는 무한을 표현할 수 없음
  - 실제 값에 매우 가깝게 표현 (근사치로 표현)

- 실수 값을 표현할 때 부호비트 s, 유효숫자 M, 지수 E를 각각 정수로 저장



수치 표현의 한계

- 불가피한 수치오류

```
>>> 0.1 + 0.1 == 0.2
True
>>> 0.1 + 0.1 + 0.1 == 0.3  # 수치오류로 인해 우리가 생각한 True가 나오지 않음
False
```

```
>>> 0.1 + 0.1 + 0.1     # 0.3이 아닌 미세한 수치오류를 가진 근사값
0.30000000000000004
```

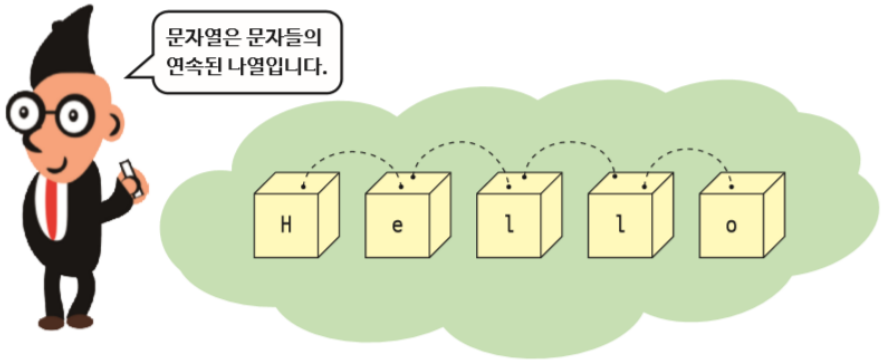
수치 표현의 한계

- 파이썬에서 정수는 높은 표현 범위를 제공

```
>>> 10 ** 100  # 10의 100승
10000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000
```

문자열

- 파이썬은 텍스트를 다루기 위한 자료형으로 문자열을 사용
  - string



문자열

- 파이썬에서는 텍스트를 큰따옴표("...")나 작은따옴표('...')로 감싸서 표현
  - "Hello Python"은 공백문자를 포함한 하나의 문자열

```
>>> "Hello Python"
'Hello Python'
```

큰따옴표로 감싸더라도 문자열로 인식하므로 표기할 적에는 작은 따옴표로 표기

문자열

- 문자열도 변수에 저장 가능

```
>>> msg = "Hello"    # 변수를 이용해서 문자열을 저장한다
>>> msg              # 문자열을 출력하면 기본적으로 작은 따옴표내에 문자가 나타남
'Hello'
>>> print(msg)       # print()로 문자열을 출력하면 따옴표는 나타나지 않음
Hello
```

  - 작은 따옴표 사용을 권장

```
>>> msg = 'Hello'
```

문자열

- 큰따옴표(")로 시작했다가 작은따옴표(')로 끝내면 문법적인 오류

```
>>> msg = "Hello'
SyntaxError: EOL while scanning string literal
```

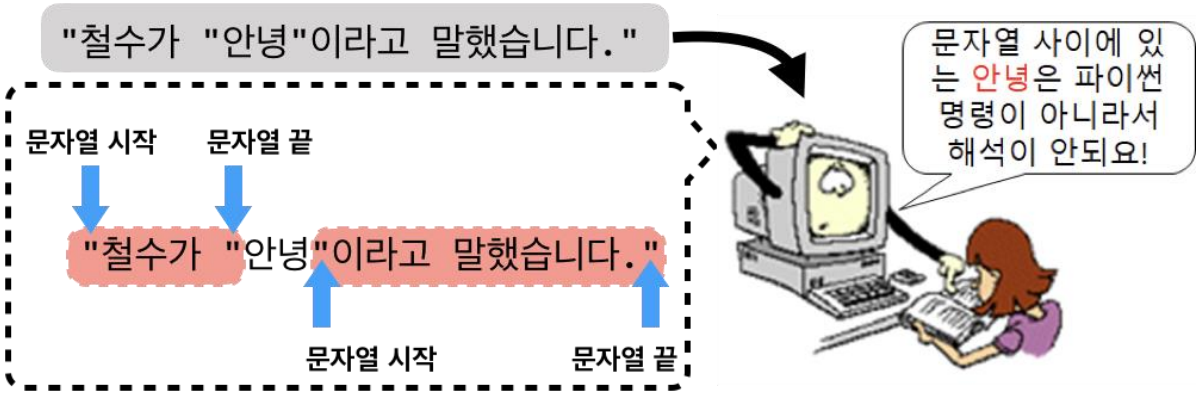
- 따옴표로 시작했는데 단어의 끝에 따옴표가 없어도 문법적인 오류

```
>>> msg = "Hello
SyntaxError: EOL while scanning string literal
```

문자열

- 왜 큰 따옴표와 작은 따옴표를 동시에 사용?

```
>>> message = "철수가 "안녕"이라고 말했습니다."
SyntaxError: invalid syntax
```



문자열

- 왜 큰 따옴표와 작은 따옴표를 동시에 사용?

```
>>> message = '철수가 "안녕"이라고 말했습니다.'
>>> message
'철수가 "안녕"이라고 말했습니다.'
>>> print(message)
철수가 "안녕"이라고 말했습니다.
```

문자열

- 문자열안에서 따옴표를 출력해야 한다면
  - ₩'
  - ₩n : 줄바꿈
  - ₩t : 탭문자
  - ₩₩ : 역슬래시

```
>>> print('철수가 \'안녕\'이라고 말했습니다.')
철수가 '안녕'이라고 말했습니다.
>>> print('안녕\n우리 다시 만나~~')
안녕
우리 다시 만나~~
>>> print('안녕\t우리 다시 만나~~')
안녕	우리 다시 만나~~
```

자료형 변환

```
>>> 100 + '원'
Traceback (most recent call last):
File "<ipython-input-3-1b3ddc5be148>", line 1, in <module>
100+"원"
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

정수(int)와 문자열(str) 사이에는
덧셈연산을 지원하지 않는다는 의미

- 자료형 통일

```
>>> str(100) + '원'    # str() 함수는 100을 문자열 '100'으로 변환시킴
'100원'
```

자료형 변환

- 정수를 문자열로 변환하려면 str()을 사용
- 문자열을 정수로 변환하려면 int()를 사용

```
>>> str(100)
'100'
>>> int("100")
100
```

- 실수 데이터를 문자열로 변환 때 str()을 사용
- 문자열을 실수로 변환하려면 float()를 사용

```
>>> str(3.14)
'3.14'
>>> float("3.14")
3.14
```

문자열 입력받기

- 사용자로부터 무언가를 입력받으려면 input()

```
input_string = input(output_string)
```

  - output_string에 담긴 문자열이 화면에 출력 후 사용자의 입력을 기다리는 상태가 됨
  - 사용자가 입력 후 엔터키를 누르면 입력된 문자들을 하나의 문자열로 반환하여 input_string에 저장

```
>>> name = input("이름을 입력하시오: ")
이름을 입력하시오: 홍길동
```

문자열 입력받기

```
이름을 입력하시오: 홍길동
홍길동 씨, 안녕하세요?
파이썬 프로그래밍의 세계에 오신 것을 환영합니다.
```

```
name = input("이름을 입력하시오: ")
print(name, "씨, 안녕하세요?")
print("파이썬 프로그래밍의 세계에 오신 것을 환영합니다.")
```

```
첫 번째 정수를 입력하시오: 300
두 번째 정수를 입력하시오: 400
300 과 400 의 합은 700 입니다.
```

```
x = int(input("첫 번째 정수를 입력하시오: "))
y = int(input("두 번째 정수를 입력하시오: "))
s = x + y
print(x, "과", y, "의 합은", s, "입니다.")
```

야구 기사를 보면 거의 비슷한 기사가 되풀이 된다. 이긴 팀이나 진 팀, 점수, 경기장, 우수 선수 등의 핵심 요소를 제외한 나머지 부분은 크게 바뀌지 않는다. 기사의 틀을 만들어두고 핵심 요소는 변수로 만들면 자동으로 기사를 작성할 수 있다. 이것을 프로그램을 만들어보자. 사용자에게 경기장, 점수, 이긴 팀과 진 팀, 우수 선수를 질문하고 변수에 저장한다. 이들 문자열에 문장을 붙여서 기사를 작성한다.

원하는 결과

```
경기장은 어디입니까? 강릉
이긴 팀은 어디입니까? 드림즈
진 팀은 어디입니까? 비룡스
우수선수는 누구입니까? 강두기
스코어는 몇대몇입니까? 9:7

==========================================
오늘 강릉 에서 야구 경기가 열렸습니다.
드림즈 과 비룡스 은 치열한 공방전을 펼쳤습니다.
강두기 의 맹활약으로  드림즈 가 비룡스 를  9:7 로 이겼습니다.
==========================================
```

```
# 사용자의 대답을 변수에 저장한다.
stadium = input("경기장은 어디입니까? ")
winner = input("이긴 팀은 어디입니까? ")
loser = input("진 팀은 어디입니까? ")
vip = input("우수선수는 누구입니까? ")
score = input("스코어는 몇대몇입니까? ")

# 사용자의 입력을 바탕으로 기사를 작성한다.
print("")
print("==========================================")
print("오늘", stadium, "에서 야구 경기가 열렸습니다.")
print(winner, "과", loser, "은 치열한 공방전을 펼쳤습니다.")
print(vip, "의 맹활약으로 ", winner,"가", loser,"를 ", score,"로 이겼습니다.")
print("==========================================")
```