고급통계프로그래밍 Solution 1

컴퓨터과학부 2017920054 이호준

과제 index는 번역본 기준입니다.

Ex 2.3

```
width = 17
height = 12.0
delimiter = '.'
```

Solution 1

- int type과 int type의 연산이나, 결과가 int형으로 표현될 수 없으므로 식의 값은 **8.5**(`float`)가 된다.

```
# Check with Code - 1
print(width/2)
print(type(width/2))

# Output
8.5
<class 'float'>
```

Solution 2

- int type과 float type의 연산이고, 이는 더 유연한 float type을 따른다. 따라서 식의 값은 **8.5**(`float`)가 된다.

```
# Check with Code - 2
print(width/2.0)
print(type(width/2.0))

# Output
8.5
<class 'float'>
```

Solution 3

- float type과 int type의 연산이고, 이는 Sol. 2와 마찬가지로 더 유연한 float type을 따른다. 따라서 식의 값은 **4.0**(`float`)이 된다.

```
# Check with Code - 3
print(height/3)
print(type(height/3))

# Output
4.0
<class 'float'>
```

Solution 4

- 모든 연산이 int형 간의 연산이고, 결과가 int형 범위에서 도출될 수 있으므로 계산 결과로 **11**(`int`)가 나온다

```
# Check with Code - 4
print(1 + 2 * 5)
print(type(1 + 2 * 5))

# Output
11
<class 'int'>
```

Solution 5

- 문자열과 int형 사이의 * 연산이므로 이는 반복 연산이 적용된다. 따라서 **.**이 5번 반복된 **.....**(`str`)을 얻을 수 있다.

```
# Check with Code - 5
print(delimiter * 5)
print(type(delimiter * 5))

# Output
.....
<class 'str'>
```

Ex 2.4

Solution 1

```
# Code for Solution 1
PI = 3.141592
r = 5
```

```
sphere_vol = (4/3) * PI * (r**3)

print(sphere_vol)

# Output
523.5986666666666
```

- π를 나타내는 변수 `PI`를 선언하고, sphere의 volume을 구하는 공식($\frac{4}{3}\pi r^3$)을 이용해 구의 부피(`sphere_vol`)을 구한 후, 이를 출력하였다.

Solution 2

```
# Code for Solution 2
copy_size = 60

book_cost = (24.95 * (4/10)) * copy_size
ship_cost = 3 + (3/4)*(copy_size-1)

total_cost = book_cost + ship_cost
print(total_cost)

# Output
646.0500000000001
```

- 사본의 수를 변수 `copy_size`로 설정하였고, 60을 할당하였다.
- 책의 총 가격을 `book_cost`로 설정하였고, 개당 서점할인가에 copy_size를 곱해 이를 구하였다.
- 운송비의 총 가격을 `ship_cost` 로 설정하였고, 주어진 운송비 공식을 이용해 이를 구하였다.
- 마지막으로 book_cost와 ship_cost를 더한 총 가격 `total_cost`를 구하고, 이를 출력하였다.
- output이 깔끔하지 못한 것은 부동소수점 오차로 인한 문제인 것으로 보인다.

Solution 3

```
# Code for Solution 3
current_sec = (6 * 3600) + 52*60

easy_pace = (8 * 60) + 15
hard_pace = (7 * 12) + 15

end_sec = current_sec + (easy_pace * 2) + (hard_pace * 3)

end_time = (end_sec // 3600, (end_sec % 3600) // 60, (end_sec % 3600) % 60)
```

```
print("운동 후 시간은 {}시 {}분 {}초입니다.".format(end_time[0], end_time[1], end_time[2]))
```

- 현재 시각을 초 단위로 변환하여 담은 변수 `current_sec` 를 선언하였다.
- 천천히 달릴 때와 빨리 달릴 때 소요되는 시간을 초로 변환한 `easy_pace` 와 `hard_pace` 를 선언하고 값을 할당해주었다.
- 운동 후 시간을 담을 변수 `end_sec` 을 선언하고, 2마일 천천히, 3마일 빨리 운동한 것을 반영하여 값을 할당해주었다.
- 종료시간을 구하기 위해 튜플 `end_time` 을 선언하고, 시, 분, 초에 맞는 값을 각각 인덱스 0, 1, 2의 값이 담을 수 있도록 하였다.
- 마지막으로 `print()`와 `.format()` 을 이용하여 각각에 해당하는 값을 출력할 수 있도록 하였다.