검색어를 입력하세요.

☰

09장 풀이

02장 파이썬 프로그래밍의 기초, 자료형

Q1.

> 홍길동 씨의 과목별 점수는 각각 다음과 같다. 홍길동 씨의 평균 점수를 구해 보자.

```
>>> a = 80
>>> b = 75
>>> c = 55
>>> (a+b+c)/3
70.0
```

Q2.

> 자연수 13이 홀수인지 짝수인지 판별할 수 있는 방법에 대해서 말해 보자.

나머지 연산자를 사용하면 자연수의 홀수, 짝수를 쉽게 판별할 수 있다

```
>>> 1 % 2
1
>>> 2 % 2
0
>>> 3 % 2
1
>>> 4 % 2
0
```

1, 2, 3, 4라는 자연수를 2로 나누었을 때의 나머지 값을 출력하는 예제이다. 결과를 보면 자연수가 홀수일 때는 1을 짝수일 때는 0을 돌려주는 것을 확인할 수 있다.

Q3.

> 홍길동씨의 주민등록번호는 881120-1068234이다. 홍길동씨의 주민등록번호를 연월일(YYYYMMDD) 부분과 그 뒤의 숫자 부분으로 나누어 출력해 보자.

```
pin = "881120-1068234"
yyyymmdd = pin[:6]
num = pin[7:]
print(yyyymmdd)  # 881120 출력
print(num)       # 1068234 출력
```

Q4.

> 주민등록번호 뒷자리의 맨 첫 번째 숫자는 성별을 나타낸다. 주민등록번호에서 성별을 나타내는 숫자를 출력해 보자.

```
pin = "881120-1068234"
print(pin[7])   # 1이면 남자, 2이면 여자
```

성별을 나타내는 숫자는 하이픈을 포함하여 8번째 숫자이므로 8번째 자리를 인덱싱한다.

Q5.

> 다음과 같은 문자열 a:b:c:d가 있다. 문자열의 replace 함수를 사용하여 a#b#c#d로 바꿔서 출력해 보자.

```
a = "a:b:c:d"
b = a.replace(":", "#")
print(b)
'a#b#c#d'
```

Q6.

> [1, 3, 5, 4, 2]라는 리스트를 [5, 4, 3, 2, 1]로 만들어보자.

```
a = [1, 3, 5, 4, 2]
a.sort( )
a.reverse( )
print(a)       # [5, 4, 3, 2, 1] 출력
```

리스트의 내장 함수인 sort를 사용하여 리스트 값들을 먼저 정렬한 후 reverse 함수를 사용하여 순서를 뒤집는다.

Q7.

> ['Life', 'is', 'too', 'short'] 라는 리스트를 Life is too short라는 문자열로 만들어 출력해 보자.

```
a = ['Life', 'is', 'too', 'short']
result = " ".join(a)
print(result)
```

a 리스트의 각 단어들을 한 문장으로 조립할 때 단어들 사이마다 공백을 넣어 주어야 한다. 1개의 공백문자(" ")를 사용하여 join한다.

Q8.

> (1,2,3)이라는 튜플에 4라는 값을 추가하여 (1,2,3,4)처럼 만들어 출력해 보자.

```
a = (1, 2, 3)
a = a + (4,)
print(a)       # (1, 2, 3, 4) 출력
```

a 튜플에 (4,)라는 튜플을 더하면 된다. 단, 이때 만들어지는 a + (4,)의 결과는 a 값이 변경되는 것이 아니라(튜플은 그 값을 변경할 수 없다) 새로운 튜플이 생성되고 그 값이 a 변수에 대입되는 것임에 유념하자.

다음 코드를 실행해 보면 a의 고유 주소 값이 변경됨을 확인할 수 있다.

```
a = (1, 2, 3)
print (id(a))     # a의 고유 주소 값 출력
a = a + (4,)
print(a)
print (id(a))     # (4,) 값이 더해 진 후 a의 고유 주소 값 출력
```

Q9.

> 다음과 같은 딕셔너리 a가 있다. 다음 중 오류가 발생하는 경우는 어떤 경우인가? 그리고 그 이유를 설명해 보자.

3번째 예를 실행하면 다음과 같은 오류가 발생한다.

```
>>> a[[1]] = 'python'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unhashable type: 'list'
```

오류가 발생하는 이유는 딕셔너리의 키로는 변하는(mutable) 값을 사용할 수 없기 때문이다. 위 예에서 키로 사용된 [1]은 리스트이므로 변하는 값이다. 다른 예에서 키로 사용된 문자열, 튜플, 숫자는 변하지 않는(immutable) 값이므로 딕셔너리의 키로 사용이 가능하다.

Q10.

> 딕셔너리 a에서 'B'에 해당되는 값을 추출해 보자.

딕셔너리도 리스트와 마찬가지로 다음과 같이 pop 함수를 사용할 수 있다.

```
a = {'A':90, 'B':80, 'C':70}
result = a.pop('B')
print(a)            # {'A':90, 'C':70} 출력
print(result)       # 80 출력
```

'B' 키 값에 해당되는 값이 리턴되고 딕셔너리 a에서는 그 값이 제거되는 것을 확인할 수 있다.

Q11.

> a 리스트에서 중복된 숫자들을 제거해 보자.

```
a = [1, 1, 1, 2, 2, 3, 3, 3, 4, 4, 5]
aSet = set(a)     # a 리스트를 집합자료형으로 변환
b = list(aSet)    # 집합자료형을 리스트 자료형으로 다시 변환
print(b)          # [1,2,3,4,5] 출력
```

리스트 자료형이 집합 자료형으로 변환되면서 중복된 값들은 사라지게 된다. 이와 같은 성질을 사용하면 리스트 내에 중 복된 값을 쉽게 제거할 수 있다.

Q12.

> 파이썬은 다음처럼 동일한 값에 여러 개의 변수를 선언할 수 있다. 다음과 같이 a, b 변수를 선언 한 후 a의 두 번째 요솟값을 변경하면 b의 값은 어떻게 될까? 그리고 이런 결과가 나오는 이유에 대해서 설명해 보자.

[1, 4, 3]이 출력된다. a와 b 변수는 모두 동일한 [1, 2, 3]이라는 리스트 객체를 가리키고 있기 때문이다.

03장 프로그램의 구조를 쌓는다! 제어문

Q1.

> 다음 코드의 결괏값은 무엇일까?

결괏값으로 shirt 가 출력된다.

1. 첫 번째 조건: "wife"라는 단어는 a 문자열에 없으므로 거짓이다.
2. 두 번째 조건: "python"이라는 단어는 a 문자열에 있지만 "you" 역시 a 문자열에 있으므로 거짓이다.
3. 세 번째 조건: "shirt"라는 단어가 a 문자열에 없으므로 참이다.
4. 네 번째 조건: "need"라는 단어가 a 문자열에 있으므로 참이다.

가장 먼저 참이 되는 것이 세 번째 조건이므로 "shirt"가 출력된다.

Q2.

> while문을 사용해 1부터 1000까지의 자연수 중 3의 배수의 합을 구해 보자.

3의 배수는 3으로 나누어 떨어지는 수이다. 이러한 아이디어를 기반으로 한 파이썬 코드는 다음과 같다.

```
result = 0
i = 1
while i <= 1000:
    if i % 3 == 0: # 3으로 나누어 떨어지는 수는 3의 배수
        result += i
    i += 1

print(result) # 166833 출력
```

Q3.

> while문을 사용하여 다음과 같이 별(*)을 표시하는 프로그램을 작성해 보자.

```
i = 0
while True:
    i += 1 # while문 수행 시 1씩 증가
    if i > 5: break     # i 값이 5이상이면 while문을 벗어난다.
    print ('*' * i)     # i 값 개수만큼 *를 출력한다.
```

while문을 수행할 때마다 i 값을 증가시킨다. 별 모양을 5번 출력해야 하므로 i 값이 5 이상일 경우 while문을 벗어나도록 한다. 별 모양을 i 값 만큼 출력하기 위해서 문자열 곱하기 기능을 사용한다.

Q4.

> for문을 사용해 1부터 100까지의 숫자를 출력해 보자.

```
>>> for i in range(1, 101):
...    print(i)
...
1
2
3
4
5
6
7
8
9
10
... 생략 ...
```

Q5.

> for문을 사용하여 A 학급의 평균 점수를 구해 보자.

```
A = [70, 60, 55, 75, 95, 90, 80, 80, 85, 100]
total = 0

for score in A:
    total += score   # A학급의 점수를 모두 더한다.

average = total / len(A) # 평균을 구하기 위해 총 점수를 총 학생수로 나눈다.
print(average) # 평균 79.0이 출력된다.
```

for문을 사용하여 먼저 총 점수를 구한 후 총 점수를 총 학생 수로 나누어 평균 점수를 구한다.

Q6.

> 리스트 중에서 홀수에만 2를 곱하여 저장하는 다음과 같은 코드가 있다.

```
numbers = [1, 2, 3, 4, 5]

result = []
for n in numbers:
    if n % 2 == 1:
        result.append(n*2)
```

위 코드를 리스트 내포(list comprehension)를 사용하여 표현해 보자.

리스트 내포로 표현하면 다음과 같다.

```
numbers = [1, 2, 3, 4, 5]
result = [n*2 for n in numbers if n%2==1]
print(result)
[2, 6, 10]
```

04장 프로그램의 입력과 출력은 어떻게 해야 할까?

Q1.

> 주어진 자연수가 홀수인지 짝수인지 판별해 주는 함수(is_odd)를 작성해 보자.

```
>>> def is_odd(number):
...     if number % 2 == 1:   # 2로 나누었을 때 나머지가 1이면 홀수이다.
...         return True
...     else:
...         return False
...
>>> is_odd(3)
True
>>> is_odd(4)
False
```

람다와 조건부 표현식을 사용하면 다음과 같이 간단하게도 만들 수 있다.

```
>>> is_odd = lambda x: True if x % 2 == 1 else False
>>> is_odd(3)
True
```

Q2.

> 입력으로 들어오는 모든 수의 평균 값을 계산해 주는 함수를 작성해 보자.

```
>>> def avg_numbers(*args):   # 입력 개수에 상관없이 사용하기 위해 *args를 사용
...     result = 0
...     for i in args:
...         result += i
...     return result / len(args)
...
>>> avg_numbers(1, 2)
1.5
>>> avg_numbers(1,2,3,4,5)
3.0
```

Q3.

> 3과 6을 입력했을 때 9가 아닌 36이라는 결괏값을 돌려주었다. 이 프로그램의 오류를 수정해 보자.

```
input1 = input("첫 번째 숫자를 입력하세요:")
input2 = input("두 번째 숫자를 입력하세요:")

total = int(input1) + int(input2)      # 입력은 항상 문자열이므로 숫자로 바꾸어 주어야 한다.
print("두수의 합은 %s 입니다" % total)
```

출력 결과는 다음과 같다.

```
첫 번째 숫자를 입력하세요:3
두 번째 숫자를 입력하세요:6
두 수의 합은 9 입니다
```

Q4.

> 다음 중 출력 결과가 다른 것 한 개를 고르시오.

```
>>> print("you" "need" "python")
youneedpython
>>> print("you"+"need"+"python")
youneedpython
>>> print("you", "need", "python")   # 콤마가 있는 경우 공백이 삽입되어 더해진다.
you need python
>>> print("".join(["you", "need", "python"]))
youneedpython
```

Q5.

> 이 프로그램은 우리가 예상한 "Life is too short"라는 문장을 출력하지 않는다. 우리가 예상한 값을 출력할 수 있도록 프로그램을 수정해 보자.

문제의 예와 같이 파일을 닫지 않은 상태에서 다시 열면 파일에 저장한 데이터를 읽을 수 없다. 따라서 열린 파일 객체를 close로 닫아준 후 다시 열어서 파일의 내용을 읽어야 한다.

```
f1 = open("test.txt", 'w')
f1.write("Life is too short!")
f1.close() # 열린 파일 객체를 닫는다.

f2 = open("test.txt", 'r')
print(f2.read())
f2.close()
```

또는 다음과 같이 close를 명시적으로 할 필요가 없는 with구문을 사용한다.

```
with open("test.txt", 'w') as f1:
    f1.write("Life is too short! ")
with open("test.txt", 'r') as f2:
    print(f2.read())
```

Q6.

> 사용자의 입력을 파일(test.txt)에 저장하는 프로그램을 작성해 보자.

기존 내용을 유지하고 새로운 내용을 덧붙이기 위해서 다음과 같이 'a' 모드를 사용해야 한다.

```
user_input = input("저장할 내용을 입력하세요:")
f = open('test.txt', 'a')   # 내용을 추가하기 위해서 'a'를 사용
f.write(user_input)
f.write("\n")               # 입력된 내용을 줄 단위로 구분하기 위해 줄 바꿈 문자 삽입
f.close()
```

Q7.

> 다음과 같은 내용을 지닌 파일 test.txt가 있다. 이 파일의 내용 중 "java"라는 문자열을 "python"으로 바꾸어서 저장해 보자.

파일을 모두 읽은 후에 문자열의 replace 함수를 사용하여 java라는 문자열을 python으로 변경한 다음 저장한다.

```
f = open('test.txt', 'r')
body = f.read()
f.close()

body = body.replace('java', 'python')

f = open('test.txt', 'w')
f.write(body)
f.close()
```

05장 파이썬 날개 달기

Q1.

> 위 클래스를 상속하는 UpgradeCalculator를 만들고 값을 뺄 수 있는 minus 메서드를 추가해 보자.

다음과 같이 Calculator 클래스를 상속하는 UpgradeCalculator 클래스를 만들고 minus 메서드를 추가한다.

```
class UpgradeCalculator(Calculator):
    def minus(self, val):
        self.value -= val
```

Q2.

> 객체변수 value가 100 이상의 값은 가질 수 없도록 제한하는 MaxLimitCalculator 클래스를 만들어 보자.

Calculator 클래스를 상속하고 add 메서드를 오버라이딩하여 다음과 같은 클래스를 만든다.

```
class MaxLimitCalculator(Calculator):
    def add(self, val):
        self.value += val
        if self.value > 100:
            self.value = 100
```

Q3.

> 다음 결과를 예측해 보자.

하나.

```
>>> all([1, 2, abs(-3)-3])
False
```

`abs(-3)` 은 -3의 절댓값이므로 3이 되어 `all([1, 2, 0])` 이 되고, 리스트의 요솟값중 0이 있기 때문에 all 내장 함수의 결과는 False가 된다.

둘.

```
>>> chr(ord('a')) == 'a'
True
```

`ord('a')` 의 결과는 97이 되어 `chr(97)` 로 치환된다. `chr(97)` 의 결과는 다시 'a'가 되므로 `'a' == 'a'` 가 되어 True를 돌려준다.

Q4.

> filter와 lambda를 사용하여 리스트 [1, -2, 3, -5, 8, -3]에서 음수를 모두 제거해 보자.

음수를 제거하기 위한 filter의 함수로 lambda 함수를 다음과 같이 만들어 실행한다.

```
>>> list(filter(lambda x:x>0, [1, -2, 3, -5, 8, -3]))
[1, 3, 8]
```

Q5.

> '0xea' 라는 16진수 문자열을 10진수로 변경해 보자.

int 내장 함수를 다음과 같이 실행한다.

```
>>> int('0xea', 16)
234
```

Q6.

> map과 lambda를 사용하여 [1, 2, 3, 4] 라는 리스트의 각 요솟값에 3이 곱해진 리스트 [3, 6, 9, 12]를 만들어 보자.

입력에 항상 3을 곱하여 돌려 주는 lambda 함수를 다음과 같이 만들고 map과 조합하여 실행한다.

```
>>> list(map(lambda x:x*3, [1,2,3,4]))
[3, 6, 9, 12]
```

Q7.

> 다음 리스트의 최댓값과 최솟값의 합을 구해 보자.

리스트의 최댓값은 max, 최솟값은 min 내장 함수를 사용하여 다음과 같이 구한다.

```
>>> a = [-8, 2, 7, 5, -3, 5, 0, 1]
>>> max(a) + min(a)
-1
```

Q8.

> 5.666666666666667을 소수점 4자리까지만 반올림하여 표시해 보자.

round 내장 함수를 사용하면 다음과 같이 반올림하여 소수점 4자리까지 표시할 수 있다.

```
>>> round(17/3, 4)
5.6667
```

Q9.

> 다음과 같이 실행할 때 입력값을 모두 더하여 출력하는 스크립트
> (`C:\doit\myargv.py`)를 작성해 보자.

다음처럼 sys모듈의 argv를 사용하여 명령 행 입력값 모두를 차례로 더해 준다.

```
import sys

numbers = sys.argv[1:] # 파일 이름을 제외한 명령 행의 모든 입력

result = 0
for number in numbers:
    result += int(number)
print(result)
```

Q10.

> os 모듈을 사용하여 다음과 같이 동작하도록 코드를 작성해 보자.

다음처럼 os 모듈의 chdir을 사용하여 `C:\doit` 이라는 디렉터리로 이동한다.

```
>>> import os
>>> os.chdir("c:/doit")
```

그리고 다음처럼 os 모듈의 popen을 사용하여 시스템 명령어인 dir을 수행한다.

```
>>> result = os.popen("dir")
```

popen의 결과를 출력하기 위해 다음과 같이 수행한다.

```
>>> print(result.read())
...
abc.txt
bidusource.html
...
```

Q11.

> glob 모듈을 사용하여 `C:\doit` 디렉터리의 파일 중 확장자가 .py인 파일만 출력하는 프로그램을 작성해 보자.

다음과 같이 glob 모듈을 사용한다.

```
>>> import glob
>>> glob.glob("c:/doit/*.py")
['c:/doit/doit01.py', 'c:/doit/test.py']
```

Q12.

> time 모듈을 사용하여 현재 날짜와 시간을 다음과 같은 형식으로 출력해 보자.

time 모듈의 strftime을 사용하여 다음과 같이 작성한다.

```
>>> import time
>>> time.strftime("%Y/%m/%d %H:%M:%S")   # %Y:년, %m:월, %d:일, %H:시, %M:분, %S:초
'2018/04/05 10:56:27'
```

Q13.

> random 모듈을 사용하여 로또 번호(1~45 사이의 숫자 6개)를 생성해 보자.

random 모듈의 randint를 사용하여 다음과 같이 작성한다.

```
import random

result = []
while len(result) < 6:
    num = random.randint(1, 45)   # 1부터 45까지의 난수 발생
    if num not in result:
        result.append(num)

print(result)
```

08장 종합문제

Q1.

> 문자열 바꾸기

```
>>> a = "a:b:c:d"
>>> b = a.split(":")
>>> b
['a', 'b', 'c', 'd']
>>> c = "#".join(b)
>>> c
'a#b#c#d'
```

Q2.

> 딕셔너리 값 추출하기

딕셔너리의 get 함수를 사용하면 해당 key가 없을 경우에는 두 번째 매개변수로 전달된 default 값을 대신 돌려준다.

```
>>> a = {'A':90, 'B':80}
>>> a.get('C', 70)
70
```

위 예에서는 'C'에 해당되는 key가 없기 때문에 디폴트 값으로 전달된 70을 돌려주었다.

Q3.

리스트의 더하기와 extend 함수

리스트 a에 + 기호를 사용하는 경우에 대해서 먼저 살펴보자.

```
>>> a = [1, 2, 3]
>>> id(a)
4302429640
```

id 함수는 입력으로 받은 리스트 a의 주소 값을 돌려 준다. 현재 a라는 리스트는 4302429640이라는 주소에 저장되어 있다.

```
>>> a = a + [4,5]
>>> a
[1, 2, 3, 4, 5]
```

리스트 a에 + 기호를 사용하여 [4, 5]라는 리스트를 더해 보았다. 그리고 다시 다음과 같이 리스트 a의 주소 값을 확인해 보자.

```
>>> id(a)
4302472072
```

이전에 리스트 a가 저장되어 있던 주소와 다른 값을 돌려주는 것을 확인할 수 있다. 주소 값이 다르기 때문에 + 를 사용하면 리스트 a의 값이 변하는 것이 아니라 두 리스트가 더해진 새로운 리스트가 반환된다는 것을 확인할 수 있다.

이번에는 extend 함수를 사용해 보자.

```
>>> a = [1, 2, 3]
>>> id(a)
4302429640
```

리스트 a를 생성하고 그 주소 값을 출력해 보았다.

```
>>> a.extend([4, 5])
>>> a
[1, 2, 3, 4, 5]
```

그리고 리스트 a에 extend를 사용하여 [4, 5]라는 리스트를 더해 주었다. 그리고 다시 다음과 같이 리스트 a의 주소 값을 확인해 보도록 하자.

```
>>> id(a)
4302429640
```

+ 기호를 사용하여 더한 경우와는 달리 주소 값이 변하지 않고 그대로 유지되는 것을 확인할 수 있다.

Q4.

리스트 총합 구하기

```
A = [20, 55, 67, 82, 45, 33, 90, 87, 100, 25]

result = 0
while A:                # A 리스트에 값이 있는 동안
    mark = A.pop()      # A리스트의 가장 마지막 항목을 하나씩 뽑아냄
    if mark >= 50:      # 50점 이상의 점수만 더함
        result += mark

print(result)           # 481 출력
```

Q5.

피보나치 함수

피보나치 수열은 다음과 같은 순서로 결괏값을 반환한다.

1. fib(0) → 0 반환
2. fib(1) → 1 반환
3. fib(2) → fib(0) + fib(1) → 0 + 1 → 1 반환
4. fib(3) → fib(1) + fib(2) → 1 + 1 → 2 반환
5. fib(4) → fib(2) + fib(3) → 1 + 2 → 3 반환
6. ...

n이 0일 때는 0을 반환, 1일 때는 1을 반환한다. n이 2 이상일 경우에는 이전의 두 값을 더하여 반환한다.

재귀 호출을 사용하면 피보나치 함수를 다음과 같이 간단하게 작성할 수 있다.

```
def fib(n):
    if n == 0 : return 0          # n이 0일 때는 0을 반환
    if n == 1 : return 1          # n이 1일 때는 1을 반환
    return fib(n-2) + fib(n-1)    # n이 2 이상일 때는 그 이전의 두 값을 더하여 반환

for i in range(10):
    print(fib(i))
```

0부터 9까지의 피보나치 수열의 결괏값을 출력하여 그 값을 확인해 보았다.

Q6.

숫자의 총합 구하기

```
user_input = input("숫자를 입력하세요: ")
numbers = user_input.split(",")
total = 0
for n in numbers:
    total += int(n)    # 입력은 문자열이므로 숫자로 변환해야 한다.
print(total)
```

수행결과

```
숫자를 입력하세요: 65,45,2,3,45,8
168
```

Q7.

한 줄 구구단

```
user_input = input("구구단을 출력할 숫자를 입력하세요(2~9):")
dan = int(user_input)       # 입력 문자열을 숫자로 변환
for i in range(1, 10):
    print(i*dan, end= ' ')  # 한 줄로 출력하기 위해 줄 바꿈 문자 대신 공백 문자를 마지막에 출력한다.
```

Q8.

역순 저장

파일 객체의 readlines를 사용하여 모든 라인을 읽은 후에 reversed를 사용하여 역순으로 정렬한 다음 다시 파일에 저장한다.

```
f = open('abc.txt', 'r')
lines = f.readlines()    # 모든 라인을 읽음
f.close()

lines.reverse()          # 읽은 라인을 역순으로 정렬

f = open('abc.txt', 'w')
for line in lines:
    line = line.strip()  # 포함되어 있는 줄 바꿈 문자 제거
    f.write(line)
    f.write('\n')        # 줄 바꿈 문자 삽입
f.close()
```

Q9.

> 평균 값 구하기

```
f = open("sample.txt")
lines = f.readlines( )  # sample.txt를 줄 단위로 모두 읽는다.
f.close( )

total = 0
for line in lines:
    score = int(line)  # 줄에 적힌 점수를 숫자형으로 변환한다.
    total += score
average = total / len(lines)

f = open("result.txt", "w")
f.write(str(average))
f.close()
```

sample.txt의 점수를 모두 읽기 위해 파일을 열고 readlines를 사용하여 각 줄의 점수 값을 모두 읽어 들여 총 점수를 구한다. 총 점수를 sample.txt 파일의 라인(Line) 수로 나누어 평균 값을 구한 후 그 결과를 result.txt 파일에 쓴다. 숫자 값은 result.txt 파일에 바로 쓸 수 없으므로 str 함수를 사용하여 문자열로 변경한 후 파일에 쓴다.

Q10.

> 사칙연산 계산기

```
class Calculator:
    def init (self, numberList):
        self.numberList = numberList

    def add(self): result = 0
        for num in self.numberList:
            result += num
        return result

    def avg(self):
        total = self.add()
        return total / len(self.numberList)

cal1 = Calculator([1,2,3,4,5])
print (cal1.add())
print (cal1.avg())

cal2 = Calculator([6,7,8,9,10])
print (cal2.add())
print (cal2.avg())
```

Q11.

> ## 모듈 사용 방법

파이썬 셸에서 mymod.py 모듈을 인식하기 위해서는 다음과 같은 3가지 방법이 있을 수 있다.

1) sys 모듈 사용하기

다음과 같이 `sys.path` 에 `C:\doit` 이라는 디렉터리를 추가하면 `C:\doit` 이라는 디렉터리에 있는 mymod 모듈을 사용할 수 있게 된다.

```
>>> import sys
>>> sys.path.append("c:/doit")
>>> import mymod
```

2) PYTHONPATH 환경 변수 사용하기

다음처럼 `PYTHONPATH` 환경 변수에 `C:\doit` 디렉터리를 지정하면 `C:\doit` 디렉터리에 있는 mymod 모듈을 사용할 수 있게 된다.

```
C:\Users\home>set PYTHONPATH=c:\doit
C:\Users\home>python
>>> import mymod
```

3) 현재 디렉터리 사용하기

파이썬 셸을 mymod.py가 있는 위치로 이동하여 실행해도 mymod 모듈을 사용할 수 있게 된다. 왜냐하면 `sys.path` 에는 현재 디렉터리인 `.` 이 항상 포함되어 있기 때문이다.

```
C:\Users\home>cd c:\doit
C:\doit>python
>>> import mymod
```

Q12.

오류와 예외 처리

7이 출력된다.

1. result의 초깃값은 0이다.
2. try문 안의 `[1, 2, 3][3]` 이라는 문장 수행 시 `IndexError` 가 발생하여 `except IndexError:` 구문으로 이동하게 되어 result에 3이 더해져 3이 된다.
3. 최종적으로 finally 구문이 실행되어 result에 4가 더해져 7이 된다.
4. `print(result)` 가 수행되어 result의 최종 값인 7이 출력된다.

Q13.

DashInsert 함수

다음 프로그램의 주석문을 참고하자.

```
data = "4546793"
numbers = list(map(int, data))   # 숫자 문자열을 숫자 리스트로 변경
result = []

for i, num in enumerate(numbers):
    result.append(str(num))
    if i < len(numbers)-1:                  # 다음 수가 있다면
        is_odd = num % 2 == 1               # 현재 수가 홀수
        is_next_odd = numbers[i+1] % 2 == 1  # 다음 수가 홀수
        if is_odd and is_next_odd:          # 연속 홀수
            result.append("-")
        elif not is_odd and not is_next_odd: # 연속 짝수
            result.append("*")

print("".join(result))
```

Q14.

문자열 압축하기

먼저 입력 문자열의 문자를 확인하여 동일한 문자가 들어올 경우에는 해당 문자의 숫자 값을 증가시킨다. 만약 다른 문자가 들어올 경우에는 해당 문자의 숫자 값을 1로 초기화하는 방법을 사용하여 작성한 코드이다.

```
def compress_string(s):
    _c = ""
    cnt = 0
    result = ""
    for c in s:
        if c!=_c:
            _c = c
            if cnt: result += str(cnt)
            result += c
            cnt = 1
        else:
            cnt +=1
    if cnt: result += str(cnt)
    return result

print (compress_string("aaabbccccca"))  # a3b2c6a1 출력
```

Q15.

Duplicate Numbers

```
def chkDupNum(s):
    result = []
    for num in s:
        if num not in result:
            result.append(num)
        else:
            return False
    return len(result) == 10

print(chkDupNum("0123456789"))      # True 리턴
print(chkDupNum("01234"))           # False 리턴
print(chkDupNum("01234567890"))     # False 리턴
print(chkDupNum("6789012345"))      # True 리턴
print(chkDupNum("012322456789"))    # False 리턴
```

리스트 자료형을 사용하여 중복된 값이 있는지 먼저 조사한다. 중복된 값이 있을 경우는 False를 리턴한다. 최종적으로 중복된 값이 없을 경우 0~9까지의 숫자가 모두 사용되었는지 판단하기 위해 입력 문자열의 숫자 값을 저장한 리스트 자료형의 총 개수가 10인지를 조사하여 10일 경우는 True, 아닐 경우는 False를 리턴한다.

Q16.

모스 부호 해독

```
dic = {
    '.-':'A','-...':'B','-.-.':'C','-..':'D','.':'E','..-.':'F',
    '--.':'G','....':'H','..':'I','.---':'J','-.-':'K','.-..':'L',
    '--':'M','-.':'N','---':'O','.--.':'P','--.-':'Q','.-.':'R',
    '...':'S','-':'T','..-':'U','...-':'V','.--':'W','-..-':'X',
    '-.--':'Y','--..':'Z'
}

def morse(src):
    result = []
    for word in src.split("  "):
        for char in word.split(" "):
            result.append(dic[char])
        result.append(" ")
    return "".join(result)


print(morse('.... .  ... .-.. . . .--. ...  . .- .-. .-.. -.--'))
```

모스 부호 규칙 표를 딕셔너리로 작성한 후 입력에 해당되는 모스 부호 문자열을 먼저 단어(공백 문자 2개)로 구분한다. 그 후 단어(공백 문자 1개)를 문자로 구분하여 해당 모스 부호 값을 딕셔너리에서 찾아서 그 결괏값을 구한다.

Q17.

기초 메타 문자

보기 중 이 조건에 해당되는 것은 B이다.

다음은 위 문제의 정규식 매치 결과를 확인해 보는 파이썬 코드이다.

```
import re

p = re.compile("a[.]{3,}b")

print (p.match("acccb"))    # None
print (p.match("a....b"))   # 매치 객체 출력
print (p.match("aaab"))     # None
print (p.match("a.cccb"))   # None
```

Q18.

문자열 검색

정규식 `[a-z]+` 은 소문자로 이루어진 단어를 뜻하므로 `5 python` 문자열에서 `python` 과 매치될 것이다. 따라서 `python` 문자열의 시작 인덱스(`m.start()`)는 2이고 마지막 인덱스(`m.end()`)는 8이므로 10이 출력된다.

```
import re

p = re.compile('[a-z]+')
m = p.search("5 python")
print(m.start() + m.end())  # 10 출력
```

Q19.

그루핑

전화번호 패턴은 다음과 같이 작성할 수 있다.

```
pat = re.compile("\d{3}[-]\d{4}[-]\d{4}")
```

이 전화번호 패턴 중 뒤의 숫자 4개를 변경할 것이므로 필요한 앞부분을 다음과 같이 그루핑한다.

```
pat = re.compile("(\d{3}[-]\d{4})[-]\d{4}")
```

컴파일된 객체 pat에 sub 함수를 사용하여 다음과 같이 문자열을 변경한다.

```
import re

s = """
park 010-9999-9988
kim 010-9909-7789
lee 010-8789-7768
"""

pat = re.compile("(\d{3}[-]\d{4})[-]\d{4}")
result = pat.sub("\g<1>-####", s)

print(result)
```

Q20.

전방 탐색

.com과 .net에 해당되는 이메일 주소만을 매치하기 위해서 이메일 주소의 도메인 부분에 다음과 같은 긍정형 전방탐색 패턴을 적용한다.

```
pat = re.compile(".*[@].*[.](?=com$|net$).*$")
```

다음은 위 패턴을 적용한 파이썬 코드이다.

```
import re

pat = re.compile(".*[@].*[.](?=com$|net$).*$")

print(pat.match("pahkey@gmail.com"))
print(pat.match("kim@daum.net"))
print(pat.match("lee@myhome.co.kr"))
```

마지막 편집일시 : 2020년 5월 6일 7:38 오전

댓글 9 피드백

- **이전글** : 08장 종합문제
- **다음글** : 10장 마치며

TOP