문화 A0019

파이썬프로그래밍

김 태 완

kimtwan21@dongduk.ac.kr

오늘 배울 내용

- 퀴즈 풀이
- 리스트 복습 : [1, 2, 3, 4]
- 튜플 : (1, 2, 3)
- 딕셔너리 : {1: 박세리, 2: 박태환, 3: 박찬호}

1번 문제

(2점 : 각 1점)

- 1부터 10까지의 숫자를 각각 홀수와 짝수인지 출력해보자.

```
for i in range(1,11):
    if i%2 == 0:
        print("%d는 짝수입니다."%i)
    else:
        print("%d는 홀수입니다."%i)
```

- 실행결과

```
1는 홀수입니다.
2는 짝수입니다.
3는 홀수입니다.
4는 짝수입니다.
5는 홀수입니다.
6는 짝수입니다.
7는 홀수입니다.
8는 짝수입니다.
9는 홀수입니다.
10는 짝수입니다.
```

2번 문제

(3점 : 각 1점)

- 30 이하의 자연수 중 2, 3의 배수를 제외한 수만 출력

```
i = 1
while i <= 30:
    if i%2 == 0 or i%3 == 0:
        i += 1
        continue
    else:
        print(i)
        i += 1
```

- 실행결과

```
1
5
7
11
13
17
19
23
25
29
```

3번 문제

(5점 : 부분점수 있음)

- 3.6.9 게임은 여러 명이 같이하는 게임입니다. 게임의 규칙은 다음과 같습니다.
  - 30을 입력해서 답이 10, 33을 입력해서 답이 14가 나오면 5점
  - 만약 두번째 노란색 박스 부분에 조금 문제가 있다고 생각되면 -2점 (즉 3점)
    - while 문을 사용해도 상관없음
  - 그것도 아니고 아래 첫번째 노란색 박스 부분만 맞았으면 -4점 (즉 1점)

```
number = int(input("게임 최대 숫자를 입력해 주세요: "))
count = 0 # 박수 수

for i in range(1, number + 1):
    for j in str(i):
        if j == '3' or j == '6' or j == '9' :
            count += 1

print(count)
```

복습

- 예제 : a 리스트 값 중에서 길이가 5인 값들만 출력

```
a = ['alpha', 'bravo', 'charlie', 'delta', 'echo', 'foxtrot', 'golf',
'cat', 'school' 'hotel', 'india’]

# coding here #
print(b)
```

- 실행 결과

```
['alpha', 'bravo', 'delta', 'india’]
```

튜플

- 튜플
  - 읽기 전용의 리스트이며, 소괄호로 생성
  - 리스트와 매우 비슷하지만 값을 읽을 수만 있으며, 수정하거나 새로 값을 추가할 수 없음
  - 튜플은 중복된 원소가 있을 수 있으며, 정해진 순서가 있어 순서가 다르면 서로 다른 튜플 : $(1,2,3) \neq (1,3,2)$
- 튜플의 생성
  - 튜플은 소괄호로 생성하지만, 괄호가 없어도 무방함

```
>>> Tup1 = (10, 20, 30)
>>> print(Tup1)
(10, 20, 30)
```

```
>>> Tup2 = 10, 20, 30
>>> print(Tup2)
(10, 20, 30)
```

  - 튜플의 항목이 한 개일 때 튜플 뒤에 콤마(,)를 붙여야 함

```
>>> Tup3 = (10)
>>> print(Tup1)
10
```

```
>>> Tup4 = 10
>>> print(Tup2)
10
```

```
>>> Tup5 = (10,)
>>> print(Tup3)
(10,)
```

```
>>> Tup6 = 10,
>>> print(Tup4)
(10,)
```

튜플

- 튜플은 읽기 전용
  - 다음의 세 가지 경우는 모두 오류

```
>>> Tup1.append(40)
>>> Tup1[0] = 40
>>> del(Tup1[0])
```

  - 튜플 자체를 통째로 삭제하려면 del(튜플이름) 함수를 사용

```
>>> del(Tup1)
```

  - 리스트에서 사용한 sort( ), reverse( ) 등의 함수도 사용 불가

튜플

- 튜플 값 읽기
  - 튜플이름[인덱스] : 특정 항목에 접근

```
>>> Tup1 = (10, 20, 30, 40)
>>> print(Tup1[0])
10
>>> print(Tup1[0] + Tup1[1] + Tup1[2])
60
>>> print(Tup1[1:3])
(20, 30)
>>> print(Tup1[1:])
(20, 30, 40)
>>> print(Tup1[:3])
(10, 20, 30)
```

튜플

- 튜플 덧셈과 곱셈

```
>>> Tup1 = (10, 20, 30, 40)
>>> Tup2 = ('A', 'B’)

>>> print(Tup1 + Tup2)
(10, 20, 30, 40, 'A', 'B’)

>>> print(Tup2 * 3)
('A', 'B', 'A', 'B', 'A', 'B')
```

- 튜플 → 리스트 → 튜플

```
>>> Tup1 = (10, 20, 30, 40)

>>> list1 = list(Tup1)
>>> list1.append(50)

>>> Tup2 = tuple(list1)

>>> print(Tup2)
(10, 20, 30, 40, 50)
```

딕셔너리

- 딕셔너리
  - 2개의 쌍이 하나로 묶이는 자료구조
    - 예시 : ‘apple:사과’ 처럼 의미 있는 두 값을 연결해 구성
  - 중괄호 { }로 묶어 구성, 키(Key)와 값(Value)의 쌍으로 구성

```
딕셔너리변수 = { 키1:값1 , 키2:값2, 키3:값3, ... }
```

- 딕셔너리 생성
  - 키와 값을 사용자가 지정하는 것이지 어떤 값을 반드시 사용해야 하는 규정은 없으며, 딕셔너리는 순서가 없음
    - 생성한 순서대로 딕셔너리가 구성되어 있다는 보장 없음
  - 예시 : 키(Key)가 1, 2, 3이, 값(Value)이 'a', 'b', 'c ' 인 딕셔너리

```
>>> myDict = {1:'a', 2:'b', 3:'c’}

>>> print(myDict)
{1: 'a', 2: 'b', 3: 'c'}
```

딕셔너리

- 딕셔너리 생성 예제
  - 딕셔너리는 여러 개의 정보를 하나의 변수로 표현할 때 유용하게 사용됨
    - 교수 김태완의 정보를 딕셔너리 구조로 생성하기

키 (key)	값 (value)
학번	20231234
이름	김태완
부서	데이터사이언스전공

```
>>> Profdict = {‘학번’: ‘20231234’ , '이름' : '김태완', '부서' : '데이터사이언스전공’}

>>> print(Profdict)
{‘학번’: ‘20231234’ , '이름' : '김태완', '부서' : '데이터사이언스전공’}
```

딕셔너리

- 딕셔너리 정보 추가
  - 정보를 추가할 때는 키와 값을 쌍으로 추가해야 함
  - '딕셔너리이름[키] = 값' 형식을 사용함
  - 만약 기존 딕셔너리에 동일한 키가 있을 경우 값이 변경

- **키는 중복되지 않고 유일함**
  - 값은 중복되어도 상관없음
  - 키 값이 중복일 경우 마지막에 있는 키가 적용

키 (key)	값 (value)
학번	20231234
이름	김태완
부서	데이터사이언스전공
나이	**39**
연락처	**'010-1234-5678'**

```
>>> Profdict['나이'] = 39
>>> Profdict['연락처'] = '010-1234-5678'

>>> print(Profdict)
{'학번': '20231234' , '이름' : '김태완', '부서': '데이터사이언스전공', '나이':39, '연락처': '010-1234-5678'}
```

딕셔너리

- 딕셔너리 정보 수정

키 (key)	값 (value)
학번	20231234
이름	**홍길동**
부서	데이터사이언스전공

```
>>> Profdict = {‘학번’: ‘20231234’ , '이름' : '김태완', '부서' : '데이터사이언스전공’}

>>> Profdict[‘이름’] = ‘홍길동’

>>> print(Profdict)
{‘학번’: ‘20231234’ , '이름' : '홍길동', '부서' : '데이터사이언스전공’}
```

딕셔너리

- 딕셔너리 키와 값 삭제
  - del(딕셔너리이름[키]) 함수를 사용

```
>>> Profdict = {‘학번’: ‘20231234’ , '이름' : '김태완', '부서' : '데이터사이언스전공’}

>>> del(Profdict[‘부서’])

>>> print(Profdict)
{‘학번’: ‘20231234’ , '이름' : '김태완'}
```

- 딕셔너리 값 읽기

```
>>> Profdict = {‘학번’: ‘20231234’ , '이름' : '김태완', '부서' : '데이터사이언스전공’}

>>> print(Profdict[‘학번’])
‘20231234’
>>> print(Profdict[‘부서’])
‘데이터사이언스전공’
```

딕셔너리

- 딕셔너리 함수
  - **keys( ) : 딕셔너리의 모든 키만 뽑아서 출력**

```
>>> Profdict = {‘학번’: ‘20231234’ , ' 이름 ' : ' 김태완', '부서' : ' 데이터사이언스전공’}

>>> print(Profdict.keys())
dict_keys([‘학번’, ‘이름’, ‘부서’])

>>> print(list(Profdict.keys()))
[‘학번’, ‘이름’, ‘부서’]
```

  - **get(키) : get 함수에 key 값을 이용하여 value 값에 접근 가능**

```
>>> Profdict = {‘학번’: ‘20231234’ , ' 이름 ' : ' 김태완', '부서' : ' 데이터사이언스전공’}

>>> print(Profdict.get(‘이름’))
‘김태완’
```

딕셔너리

- 딕셔너리 함수
  - **values( ) : 딕셔너리의 모든 값만 뽑아서 출력**
    - dict_values가 보기 싫으면 list(딕셔너리이름.values()) 함수를 사용

```
>>> Profdict = {‘학번’: ‘20231234’ , ' 이름 ' : ' 김태완', '부서' : ' 데이터사이언스전공’}

>>> print(Profdict.values())
dict_keys([‘20231234’, ‘김태완’, ‘데이터사이언스전공’])

>>> print(list(Profdict.values()))
[‘20231234’, ‘김태완’, ‘데이터사이언스전공’]
```

  - **items( ) : 튜플 형태로 구하기**

```
>>> print(Profdict.items())
dict_items([(‘학번’,‘20231234’), (‘이름’,‘김태완’), (‘부서’,‘데이터사이언스전공’)])
```

딕셔너리

- 딕셔너리 함수
  - **in : 딕셔너리 안에 키가 있는지는 확인**

```
>>> Profdict = {‘학번’: ‘20231234’ , ' 이름 ' : ' 김태완', '부서': ' 데이터사이언스전공’}

>>> print(‘학번’ in Profdict)
True

>>> print(‘연락처’ in Profdict)
False
```

딕셔너리

- 딕셔너리 함수
  - 반복문을 이용하여 딕셔너리에 저장된 키와 값 모두 출력 가능

```
>>> Profdict = {‘학번’: ‘20231234’ , '이름':'김태완', '부서':'데이터사이언스전공’}

>>> for key in Profdict.keys():
        print(key, Profdict[key])

>>> for k,v in Profdict.items():
        print(k, v)
```

딕셔너리

- 딕셔너리 예제

이름	가격	재고
메로나	1000	32
비비빅	900	25
죠스바	1100	8

- 아이스크림 이름을 key 값으로 [가격, 재고]리스트를 딕셔너리 값으로 생성

```
icecream = {“메로나":[1000, 32],"비비빅":[900, 25],"죠스바":[1100, 8]}
```

- 메로나 가격 출력

```
print(icecream['메로나’][0])
```

- 비비빅 재고 출력

```
print(icecream['비비빅’][1])
```

- 월드콘 [가격 1400, 재고 12] 데이터 추가

```
icecream['월드콘’] = [1400, 12]
```

예제

- **끝 입력 전까지 무한 반복문 생성**

```
foods = {"떡볶이":"김밥", "자장면":"단무지","라면":"파김치","치킨":"맥주
","삼겹살":"소주"}

while True:
# coding here #
```

- **실행 결과**

```
['떡볶이', '자장면', '라면', '치킨', '삼겹살']중 좋아하는 음식은? 자장면
자장면 궁합 음식은 단무지 입니다.
['떡볶이', '자장면', '라면', '치킨', '삼겹살']중 좋아하는 음식은? 라면
라면 궁합 음식은 파김치 입니다.
['떡볶이', '자장면', '라면', '치킨', '삼겹살']중 좋아하는 음식은? 치킨
치킨 궁합 음식은 맥주 입니다.
['떡볶이', '자장면', '라면', '치킨', '삼겹살']중 좋아하는 음식은? 끝
```

예제

- **튜플로 관리자 정보 처리**

```
admin_info = [‘admin’, ‘12345’, ‘data@dongduk.ac.kr']

# coding here #
```

- **실행 결과**

```
👉 실행 결과 1
관리자 아이디를 입력하세요 : rubato
관리자 비밀번호를 입력하세요 : 1111
아이디 또는 비밀번호가 잘못 입력되었습니다.

👉 실행 결과 2
관리자 아이디를 입력하세요 : admin
관리자 비밀번호를 입력하세요 : 12345
관리자입니다.
```

예제

- **딕셔너리로 성적 합계/평균 구하기**

```
scores = [‘김예진’: 90, ‘박영진’: 95, ‘김소희’: 84]
sum = 0

for key in scores:
    sum += 1
    print('%s : %d' % ( 2 , scores[key]))

avg = sum/len( 3 )

print('합계 : %d, 평균 : %.2f' % (sum, avg))
```

- **실행 결과**

```
👉 실행 결과
김예진 : 90
박영진 : 95
김소희 : 84
합계 : 269, 평균 : 89.67
```

예제

- **딕셔너리로 영단어 퀴즈 만들기**

```
words = {'사과': 'apple', '컴퓨터': 'computer', '학교': 'school', '책상': 'desk', '의자': 'chair'}

# coding here #
```

- **실행 결과**

```
👉 실행 결과
사과에 해당되는 영어 단어를 입력해주세요: apple
정답입니다!
컴퓨터에 해당되는 영어 단어를 입력해주세요: commputer
틀렸습니다!
학교에 해당되는 영어 단어를 입력해주세요: school
정답입니다!
책상에 해당되는 영어 단어를 입력해주세요: tesk
틀렸습니다!
의자에 해당되는 영어 단어를 입력해주세요 | chaaar
틀렸습니다!
```

감사합니다

kimtwan21@dongduk.ac.kr

김 태 완