Lec_04

Variables 3

Function 1

Loop Syntax 2

Variables 3

- Dictionary
  - 데이터 전체를 중괄호({})로 묶어 사용
  - 하나의 데이터는 Key와 Value로 구성되어 있음
  - Key는 숫자나 문자열
  - Value는 다양한 자료형을 사용할 수 있음

```
dict = { key1:value1, key2:value2, key3:value3 }

menu = { 1:'한식', 2:'중식', 3:'양식'}
menu_c = { '짬뽕':8000, '짜장면':6000, '탕수육':12000 }
```

Variables 3

- Dictionary
  - Key를 이용해 Value에 접근

```
print(menu[2])
print(menu['짬뽕'])          // 잘못된 Key인 경우 에러발생!
print(menu.get('짬뽕'))      // 잘못된 Key인 경우에도 에러 발생하지 않음!
```

  - Value에 다양한 데이터 형식을 사용해 활용

```
menu_k = {'한식':['김치찌개', '된장찌개', '비빔밥']}
menu_kp = {'한식':{'김치찌개':6000, '된장찌개': 6000, '비빔밥': 5000}}
```

Variables 3

- Dictionary

```
del menu[2]                         // Key에 해당하는 자료 삭제

print(menu.keys())                  // 딕셔너리의 전체 Key를 리스트 형태로 반환
print(menu.values())                // 딕셔너리의 전체 Value를 리스트 형태로 반환
print(menu.items())                 // 딕셔너리의 전체 Key와 Value 쌍을 튜플 형태로 반환

menu_add = {4:'분식'}
menu.update(menu_add)               // 딕셔너리에 딕셔너리 형식의 자료 추가
print(menu)

menu.clear()                        // 딕셔너리의 모든 자료 삭제
```

Function 1

- Parameter와 Return이 없는 함수
  - 반복되는 코드를 하나의 함수로 정의
  - 코드를 간결하고 보기 좋게
  - 함수의 이름으로 내용을 유추할 수 있게

```
def func_1():                                   // 함수 정의
    print("======< 메 뉴 >======")
    print("    1. 한  식")
    print("    2. 중  식")
    print("    3. 양  식")

func()                                          // 함수 사용
```

Function 1

- Parameter는 있고 Return이 없는 함수
  - 인자를 받아서 함수 내에서 처리
  - 함수 내 처리 결과를 반환하지 않음
  - 인자를 받아 출력처리하는 등으로 활용

```
def func_2(dict):                    // 함수 정의
    print(dict[1])
    print(dict[2])
    print(dict[3])
    print(dict[4])

func_2(menu)                         // 함수 사용
```

Function 1

- Parameter와 Return이 모두 있는 함수
  - 인자를 받아 처리한 결과를 return <변수명> 형식으로 반환
  - 전형적인 y=f(x) 구조

```
def func_3(a, b):
    sum = a + b
    mul = a * b
    return sum, mul

c, d = func_3(a, b)
print('덧셈: {}, 곱셈: {}'.format(c, d))
```

Loop Syntax 2

- For 문
  - **프로그래밍에서 가장 중요한 문법 중 하나**
  - 지정한 범위 동안 코드 블록을 반복
  - 반복 범위는 리스트 혹은 range()를 사용

```
for i in [0, 1, 2, 3, 6, 7]:
    print(i)

for i in range(0, 20, 2):              // range(start, end, step)
    print(i)
```

Loop Syntax 2

- For 문
  - 반복가능한(iterable) 자료형을 범위로 지정 가능

```
student = ['Nam', 'Song', 'Jung', 'Kim']

for name in student:
    print(name)

for k in menu.keys():                   // 딕셔너리의 key들만 차례대로 꺼냄
    print(k)

for k, v in menu.items():               // 딕셔너리의 key와 value를 차례대로 꺼냄
    print('{}:{}'.format(k, v))
```

Loop Syntax 2

- 연습문제
  - 중복 for문을 사용하여 2단 ~ 9단 표현

```
2 x 1 = 2       3 x 1 = 3       4 x 1 = 4 ...
2 x 2 = 4       3 x 2 = 6       4 x 2 = 8 ...
2 x 3 = 6       3 x 3 = 9       4 x 3 = 12 ...
```

[MiniProject] 메뉴판 업그레이드 2

- 함수를 활용하여 중복 구문을 정리
- 딕셔너리를 활용하여 각 메뉴의 이름과 가격을 저장
- for문을 활용하여 메뉴 출력 시 저장한 메뉴를 하나씩 출력
- 마지막에는 총 합계 금액을 출력
- (옵션1) 손님으로부터 금액을 받아 거스름돈을 계산하여 출력
- (옵션2) 계산 후 대기번호를 출력