1주차-2

1. 오늘은 무엇을 배우는가?

1. 함수와 클래스
    a. 코드의 중복을 줄여주는 함수에 대해서 배웁니다.
    b. 함수를 담는 바구니인 클래스에 대해 배웁니다.
    c. 함수와 클래스를 왜 써야 하는지.
    d. 함수와 클래스를 어떻게 선언하는지.
    e. 클래스로부터 메소드를 어떻게 호출하는 지를 배웁니다.
    f. 심화
        i. 네임스페이스 : 왜 self 파라미터를 클래스 함수에서 필요로 하는지 배웁니다.
2. 객체 지향
    a. 함수와 클래스로 구현하는 객체 지향에 대해 다룹니다.
3. 파일 읽고 쓰기
    a. 파일을 열고open, 읽고read, 쓰고write, 추가하는append 방법에 대해 배웁니다.
4. 모듈
    a. import가 어떤 기능을 하는지 살펴봅니다.

2. 함수

- 함수
    - 입력에 대해 어떤 작업을 수행하여 출력을 제공하는 도구

- 왜 쓰는가?
  - **반복되는 코드를 줄이기 위해서입니다.**
- 반복되는 코드를 줄이려는 이유가 무엇인가?
  - 코드를 효율적으로 작성하고 관리하기 위해서입니다.
- 함수의 종류
  - 파라미터가 있는, 없는 함수
    - add(a, b)에서 괄호 안의 a, b가 파라미터.

```
def 안녕():
    print("안녕하세요!")

def 인사하기(이름):
    print("{}님 안녕하세요!".format(이름))

안녕() # 안녕하세요!
인사하기("홍길동") # 홍길동님 안녕하세요!
```

  - 반환값이 있는, 없는 함수
    - return 문이 존재하면 반환값이 있는 것.

```
def print_add(a, b):
  print(a + b)

def add(a, b):
  return a + b

print_add(1, 2) # 3
c = add(1, 2) # 반환받을 변수를 지정해주어야 함
print(c) # 3
```

3. 클래스

- 클래스
  - **변수와 함수를 담은 바구니**
- 왜 쓰는가?
  - 어떤 변수, 함수들은 하나의 그룹으로 묶일 수 있습니다. 그런 함수들의 관리를 편하게 하기 위해서입니다.

    - 학생증은 '이름, 전공, 잔액' 등의 변수를 가지고, '결제하기' 같은 함수를 가집니다. 이런 변수, 함수를 한 데 묶어둔 다면 나중에 많은 학생이 들어오더라도 쉽게 추가할 수 있게 됩니다.

- 예제 코드

```
class 학생증():
    def __init__(self, 이름, 전공, money):
        self.name = 이름
        self.major = 전공
        self.money = money

    def 학생_소개(self):
        print("이름은 {}입니다.".format(self.name))
        print("전공은 {}입니다.".format(self.major))
        print("남은 금액은 {}원입니다.".format(self.money))

    def 결제하기(self, cost):
        self.money = self.money - cost
        print("결제되었습니다.")
        print("남은 금액은 {}원입니다.".format(self.money))

홍길동_학생증 = 학생증("홍길동", "전기", 10000)
홍길동_학생증.학생_소개()
"""
이름은 홍길동입니다.
전공은 전기입니다.
남은 금액은 10000원입니다.
"""

홍길동_학생증.결제하기(1000)
"""
결제되었습니다.
남은 금액은 9000원입니다.
"""
```

3.1. 질문 : 저 self는 대체 뭐죠?

```
def 결제하기(self, cost): # 함수에서는 파라미터를 두 개 선언했는데

홍길동_학생증.결제하기(1000) # 함수를 쓸 때는 변수 하나만 넘긴다?
```

- self의 정체를 알아내기 위해 네임스페이스 개념을 알아야 합니다.
- 네임스페이스는 **변수나 함수를 찾는 우선순위**라고 생각하면 됩니다.
- 같은 이름의 함수가 두 개 있을 경우, 개발자가 그 함수를 부르면 파이썬은 어떤 걸 선택해야 할까요?

```
# keri.py
def 안녕():
  print("안녕~~")

# main.py
from keri import *

def 안녕():
  print("반가워~~")

안녕() # 반가워~~
```

- 위 코드 예시처럼, 가장 먼저 접근할 수 있는 함수부터 반환하게 되며, 이 우선순위를 결정해놓은 게 네임스페이스입니다.
- self 파라미터는 함수가 사용될 때마다 이 함수의 소속이 어디인지 시스템에 넘겨줍니다(유저가 아니라).
    - 즉 홍길동_학생증.결제하기(1000)를 부르면 결제하기()라는 함수는 “난 ‘학생증’이라는 클래스 소속의 함수입니다”라는 정보를 시스템에 준다는 의미입니다.

4. 객체 지향 프로그래밍

- 객체 지향 프로그래밍은 클래스, 함수 등의 객체 개념을 사용하여 효율적인 개발을 추구하는 프로그래밍 모델입니다.
    - **객체 지향 프로그래밍의 목적은 효율적이고, 협업에 유리한 코드의 작성입니다.**
- 그럼 객체 지향 프로그램이 아니면 무슨 지향 프로그램인가요?
    - C언어처럼 int main(void) 아래에 모든 기능을 다 넣으면 절차 지향 프로그래밍이 됩니다.
    - 함수로 기능을 잘 나누어서 기능을 함수 호출로 처리하면 함수 지향 프로그래밍이 됩니다.

5. 파일 읽고 쓰기

- 파일 읽기

학생정보.txt

홍길동
전기
10000

```
with open("학생정보.txt", "r") as f:
  내용 = f.read()

print(내용)

"""
홍길동
전기
10000
"""
```

```
with open("학생정보.txt", "r") as f:
    내용리스트 = f.readlines()

print(내용리스트)

"""
['홍길동\n', '전기\n', '10000']
"""
```

```
# \n 부분을 제거하기 위해 strip() 함수를 사용할 수 있습니다.

for i in range(len(내용리스트)):
    내용리스트[i] = 내용리스트[i].strip()

print(내용리스트)

"""
['홍길동', '전기', '10000']
"""
```

- 파일 쓰기, 추가하기

```
# 4.2. 쓰기

내용 = """1주차-2 내용입니다.
함수, 클래스에 대해 다룹니다."""
```

```
with open("내용요약.txt", "w") as f:
    f.write(내용)

추가할_내용 = "\n파일 쓰기, 모듈 역시 다룹니다."
with open("내용요약.txt", "a") as f:
    f.write(추가할_내용)
```

6. 모듈

```
# keri.py

def 안녕():
    print("안녕~~")
```

- import를 통해 외부 파이썬 파일의 함수를 활용할 수 있습니다.

```
import keri

keri.안녕() # 안녕~~
```

- from, import를 통해 특정 함수만 가져올 수 있습니다.

```
from keri import *

안녕() # 안녕~~
```