파이썬 입문

한국폴리텍대학

2023.07.13

Python 복습

1. 모듈 만들기 # 패키지 만들기. package name: sports

. 모듈 : tennis, golf, soccer

. 각 모듈을 구성하는 요소

- tennis : 변수 1개(t=100), 함수 1개(func_t : print('tennis')), 클래스 1개(c_t : 속성 t = 100, 메소드1개로 구성)
- golf : 변수 1개(g=200), 함수 1개(func_g : print('golf')), 클래스 1개(c_g : 속성 g = 200, 메소드1개로 구성)
- soccer : 변수 1개(s=300), 함수 1개(func_s : print('soccer')), 클래스 1개(c_s : 속성 s = 300, 메소드1개로 구성)

. test.py를 만들고

- 각 모듈에 있는 변수 출력 (print)
- 각 모듈에 있는 함수 호출 (invoke, call)
- 각 모듈에 있는 클래스의 객체를 생성한 후, 메소드를 이용하여 속성을 출력 (메소드에 print 포함 또는 리턴값을 출력)

Python 복습

1. calc 리스트를 만들고, 더하기 함수, 빼기 함수, 곱하기 함수, 나누기 함수를 리스트 안에 저장하고 사용해 보세요.

   더하기 함수 이름: my_add, 빼기 함수 이름: my_subtract, 곱하기 함수 이름: my_multiply, 나누기 함수 이름: my_divide

2. calc_d 딕셔너리를 만들고, 더하기 함수, 곱하기 함수, 나누기 함수를 dict 안에 저장하고 사용해 보세요.

   더하기 함수 이름 (key: 0 , value: my_add)

   빼기 함수 이름 (key: 1 , value: my_subtract)

   곱하기 함수 이름 (key: 2 , value: my_multiply)

   나누기 함수 이름 (key: 3 , value: my_divide)

Python 복습

1. calc 리스트를 만들고, 더하기 클래스, 빼기 클래스, 곱하기 클래스, 나누기 클래스를 리스트 안에 저장하고 사용해 보세요.

   더하기 클래스 이름: my_add, 빼기 클래스 이름: my_subtract, 곱하기 클래스 이름: my_multiply, 나누기 클래스 이름: my_divide

   각 클래스의 계산하는 메소드는 __call__ 를 사용하세요.

2. calc_d 딕셔너리를 만들고, 더하기 클래스, 빼기 클래스, 곱하기 클래스, 나누기 클래스 를 dict 안에 저장하고 사용해 보세요.

   더하기 클래스 이름 (key: 0 , value: my_add)

   빼기 클래스 이름 (key: 1 , value: my_subtract)

   곱하기 클래스 이름 (key: 2 , value: my_multiply)

   나누기 클래스 이름 (key: 3 , value: my_divide)

Python 복습

1. List Comprehension을 이용하여 1, 3, 5, 7, … , 97, 99 요소를 갖는 리스트 data를 만드세요.

   data = [1, 3, 5, … , 97, 99]

2, map과 lambda를 이용하여 data의 모든 요소에 10을 더하고, float형으로 변환하세요.

   data = [11.0, 13.0, 15.0, … , 107.0, 109.0]

1. List Comprehension을 이용하여 1부터 45까지 요소를 갖는 리스트 lotto를 만드세요.

   lotto = [1, 2, 3, … , 44, 45]

2. lotto 요소의 순서를 섞고, 맨 앞 6개의 요소를 출력하세요.

   lotto = [27, 14, 3, 9, … , 33, 8]

   print로 출력 27, 14, 3, 9, …

Python 복습

1. Person 클래스를 만드세요.

   . 속성은 age, name

2. Person을 상속받아 Korean 클래스를 만드세요.

   . 속성은 city

3. Korean을 상속받아 Student 클래스를 만드세요.

   . 속성은 subject

4. 3명의 학생을 만들고 각 학생의 age, name, city, subject

   를 출력하세요.

   나이(age) : 예 30, 25, 40

   이름(name) : 예 kim, lee, park

   도시(city) : 예 Asan, Seoul, Jeju

   과목(subject) : 예 Python, Java, AI

super()

```
#  super()              :  부모클래스를 가리킨다

    super().__init__()  :  self 인자가 없다
    super().run()
    super().test()

#  super(A, self)      :  자식클래스 A의 부모클래스를 가리킨다

#  class D(C, B, A):        # 다중상속의 경우 super()는 부모클래스들을 순서대로 가리킨다
          def __init__(self):
                super().__init__()    --> C
                # super(D, self).__init__()    --> C
                # super(C, self).__init__()    --> B
                # super(B, self).__init__()    --> A
                # super(A, self).__init__()    --> object
```

super()

```
class A():
    def __init__(self):
        print('A start')
        super().__init__()
        print('A end')
    def test(self):
        print('A')

class B():
    def __init__(self):
        print('B start')
        super().__init__()
        print('B end')
    def test(self):
        print('B')
```

```
class C():
    def __init__(self):
        print('C start')
        super().__init__()
        print('C end')
    def test(self):
        print('C')

class D(C, B, A):
    def __init__(self):
        print('D start')
        super().__init__()
        # super().test()                # C
        # super(B, self).test()         # A
        # B.test(self)                  # B   -->  self 인자 있음
        print('D end')

result = D()            #  D C B A A B C D
result.test()           #  C

print(D.__mro__)       # Method Resolution Order (MRO) : 메소드 결정 순서
```

```
class A():
    def __init__(self):
        self.a = 100
class B(A):
    def __init__(self):
        print(self.a)       # self.a는 없다. error
```

super()

```
class A():
    def __init__(self, a):
        print('A start')
        super().__init__()
        print('A end')
    def test(self):
        print('A')

class B():
    def __init__(self, a):
        print('B start')
        super().__init__()
        print('B end')
    def test(self):
        print('B')
```

```
class C():
    def __init__(self, a, b):
        print('C start')
        super().__init__()
        print('C end')
    def test(self):
        print('C')

class D(C, B, A):
    def __init__(self):
        print('D start')
        # super().__init__(30, 'a') # C클래스만 초기화된다
        # 각 부모클래스의 초기화가 필요한 경우
        # C.__init__(self, 30, 'a')   # 또는 super(D, self).__init(30, 'a')
        # B.__init__(self, 'b')       # 또는 super(C, self).__init('b')
        # A.__init__(self, 5)         # 또는 super(B, self).__init(5)
        print('D end')

result = D()          #  D C B A A B C D
result.test()         #  C

print(D.__mro__)      # Method Resolution Order (MRO) : 메소드 결정 순서
```

MRO

```
class H():
    def __init__(self):
        print('H start')
    def test(self):
        print('H')

class A():
    def __init__(self):
        print('A start')
        print('A end')
    def test(self):
        print('A')

class B(H):
    def __init__(self):
        print('B start')
        # super().test()     # D C C B H B
        print('B end')
    def test(self):
        print('B')
```

```
class C(H):
    def __init__(self):
        print('C start')
        super().test()              #  D C B C B B
        print('C end')
    def test(self):
        print('C')

class D(C, B, A):
    def __init__(self):
        print('D start')
        super(D, self).__init__()
        super(C, self).__init__()

result = D()

print(D.__mro__)      # D C B H A object
```

파이썬 입문

끝