LAB

```python
class Television:
    def __init__(self, channel, volume, on):
        self.channel = channel
        self.volume = volume
        self.on = on

    def show(self):
        print(self.channel, self.volume, self.on)

    def setChannel(self, channel):
        self.channel = channel

    def getChannel(self):
        return self.channel
t = Television(9, 10, True)
t.show()

t.setChannel(11)
t.show()
```

```
9 10 True
11 10 True
```

```python
import math

class Circle:
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return math.pi * self.radius ** 2

    def circumference(self):
        return 2 * math.pi * self.radius

c= Circle(10)
print("원의 면적",c.area())
print("원의 둘레",c.circumference())
```

4] ✓ 0.0s

```
원의 면적 314.1592653589793
원의 둘레 62.83185307179586
```

```python
class Car:
    def __init__(self, speed, color, model):
        self.speed = speed
        self.color = color
        self.model = model
    def drive(self):
        self.speed = 60
myCar = Car(0, "blue", "E-class")

print("자동차 객체를  생성하였습니다.")
print("자동차의 색상은", myCar.color, "입니다.")
print("자동차의 모델은", myCar.model, "입니다.")
print("자동차의 속도는", myCar.speed, "입니다.")

myCar.drive()
print("자동차의 속도를", myCar.speed, "로 변경하였습니다.")
```

5] ✓ 0.0s

```
자동차 객체를  생성하였습니다.
자동차의 색상은 blue 입니다.
자동차의 모델은 E-class 입니다.
자동차의 속도는 0 입니다.
자동차의 속도를 60 로 변경하였습니다.
```

```python
class student:
    def __init__(self, name, age, major):
        self.__name = name
        self.__age = age
        self.__major = major

obj= student("홍길동", 20, "컴퓨터")
print(obj.__dict__)
print(obj._student__name) # name
print(obj._student__age) # age
print(obj._student__major) # major
```

[18] ✓ 0.0s

```
{'_student__name': '홍길동', '_student__age': 20, '_student__major': '컴퓨터'}
홍길동
20
컴퓨터
```

```python
class Student:
    def __init__(self, name=None, age=0):
        self.__name = name
        self.__age = age
    def getAge(self):
        return self.__age # 접근자 메소드
    def getName(self):
        return self.__name # 접근자 메소드드
    def setAge(self, age):
        self.__age=age # 설정자 메소드
    def setName(self, name):
        self.__name=name # 설정자 메소드
obj=Student("Hong", 20)
obj.getName()
obj.getAge()
```

✓ 0.0s

```
20
```

```python
class 은행계좌:
    def __init__(self):
        self.__현재_잔액 = 0
    def 입금(self, 금액):
        self.__현재_잔액 += 금액
        print("통장에서", 금액, "원이 입금되었습니다.")
    def 출금(self, 금액):
        self.__현재_잔액 -= 금액
        print("통장에서", 금액, "원이 출금되었습니다.")

a = 은행계좌()
b= int(input("입금할 금액을 입력하세요: "))
c= int(input("출금할 금액을 입력하세요: "))
a.입금(b)
a.출금(c)
print("현재 잔액은", a._은행계좌__현재_잔액, "원입니다.")
```

✓ 6.3s

```
통장에서 10000 원이 입금되었습니다.
통장에서 5000 원이 출금되었습니다.
현재 잔액은 5000 원입니다.
```

```python
class Dog:
    kind = "Bulldog"            # 클래스변수
    def __init__(self, name, age):
        self.name = name     # 각인스턴스에유일한인스턴스변수
        self.age = age     # 각인스턴스에유일한인스턴스변수
```

0.0s

```python
class Vector2D :
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __add__(self, other):
        return Vector2D(self.x + other.x, self.y + other.y)

    def __sub__(self, other):
        return Vector2D(self.x - other.x, self.y - other.y)

    def __eq__(self, other):
        return self.x == other.x and self.y == other.y

    def __str__(self):
        return f'({self.x}, {self.y})'

u = Vector2D(0,1)
v = Vector2D(1,0)
w = Vector2D(1,1)

a = u + v       # (1, 1)

print(a)

print(a == w)
```

✓ 0.0s

```
(1, 1)
True
```

실습

```python
class Rectangle:

    def __init__(self, x, y, w, h):
        self.x = x  # 최측 상단 x좌표
        self.y = y  # 최측 상단 y좌표
        self.w = w  # 너비
        self.h =h  # 높이

    def __str__(self):
        return f"Rectangle at ({self.x}, {self.y}) with w {self.w} and h {self.h} and area is {self.w*self.h}"

    # 접근자 (getter)
    def getX(self):
        return self.x

    def getY(self):
        return self.y

    def getWidth(self):
        return self.w

    def getHeight(self):
        return self.h

    # 설정자 (setter)
    def setX(self, x):
        self.x = x

    def setY(self, y):
        self.y = y

    def setWidth(self, w):
        self.w = w

    def setHeight(self,h):
        self.h =h

    # 면적 계산
    def getArea(self):
        return self.w * self.h

    # 겹침 여부 확인
    def Overlap(self, r):
        # 두 사각형이 겹치지 않는 경우를 판별
        if (self.x + self.w <= r.x or r.x + r.w <= self.x or
            self.y + self.h <= r.y or r.y + r.h <= self.y):
            return False
        return True
r1 = Rectangle(0, 0, 100, 100)
r2 = Rectangle(10, 10, 100, 100)
print(r1.Overlap(r2))
```

[6] ✓ 0.0s

```
True
```

```python
class PhoneBook:
    def __init__(self):
        self.contacts = {}  # 연락처 저장용 딕셔너리 초기화

    def __str__(self):
        if not self.contacts:
            return "연락처가 없습니다."
        result = ""
        for name, info in self.contacts.items():
            result += f"{name}\n"

            result += f"office phone: {info.get('office', '없음')}\n"
            result += f"email address: {info.get('email', '없음')}\n"
        return result

    def add(self, name, mobile=None, office=None, email=None):
        self.contacts[name] = {
            'mobile': mobile,
            'office': office,
            'email': email
        }

obj = PhoneBook()
obj .add("Kim" , office="1234567", email="kim@company.com" )
obj .add("Park" , office="2345678", email="park@company.com")
print(obj)
```

✓ 0.0s

```
Kim
office phone: 1234567
email address: kim@company.com
Park
office phone: 2345678
email address: park@company.com
```