```
# 문제 1

class Box:
    def __init__(self, l, h, d):
        self.length = l
        self.height = h
        self.depth = d

    def __str__(self):
        return f"가로: {self.length}, 세로: {self.height}, 높이: {self.depth}"

    def setLength(self, l):
        self.length = l

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

    def setDepth(self, d):
        self.depth = d

    def volume(self):
        self.volume = self.length * self.height * self.depth
        return self.volume

b1 = Box(100, 100, 100)

print(b1.volume())

b1.setDepth(4)
b1.setHeight(3)
b1.setLength(8)

print(b1)
```

```
1000000
가로: 8, 세로: 3, 높이: 4
```

```
# 문제 2
# 강아지를 나타내는 클래스 Dog작성
# 이름, 나이, 색상 속성을 가짐
class Dog:
    count = 0

    def __init__(self, name, age, color):
        self.name = name
        self.age = age
        self.color = color
        Dog.count += 1

    def __str__(self):
        return f"이름: {self.name}, 나이: {self.age}, 색: {self.color}"

    def dog_num(self):
        print(f"지금까지 생성된 강아지의 수 = {Dog.count}")

b1 = Dog("Molly", 10, "brown")
b2 = Dog("Daisy", 6, "black")
b3 = Dog("Bella", 7, "white")

b1.dog_num()
b2.dog_num()
b3.dog_num()
print(Dog.count)
```

```
지금까지 생성된 강아지의 수 = 3
지금까지 생성된 강아지의 수 = 3
지금까지 생성된 강아지의 수 = 3
3
```

```
# 문제 3
# 마술사에게 보내면 강아지의 나이를 5살 어리게 만든다
# 마술사를 나타내는 클래스 Witch는  younger() 메소드를 가짐

# 이전에 생성한 Dog()클래스
class Dog:
    count = 0

    def __init__(self, name, age, color):
        self.name = name
        self.age = age
```

```
        self.color = color
        Dog.count += 1

    def __str__(self):
        return f"이름: {self.name}, 나이: {self.age}, 색: {self.color}"

    def dog_num(self):
        print(f"지금까지 생성된 강아지의 수 = {Dog.count}")

# Witch 클래스
class Witch:
  def younger(self, dog):
    dog.age -= 5

witch = Witch()

b1 = Dog("Molly", 10, "brown")
b2 = Dog("Daisy", 6, "black")
b3 = Dog("Bella", 7, "white")

print(b1)
print(b2)
print(b3)

print("Witch한테 보내기")
witch.younger(b1)
witch.younger(b2)
witch.younger(b3)

print(b1)
print(b2)
print(b3)
```

```
이름: Molly, 나이: 10, 색: brown
이름: Daisy, 나이: 6, 색: black
이름: Bella, 나이: 7, 색: white
Witch한테 보내기
이름: Molly, 나이: 5, 색: brown
이름: Daisy, 나이: 1, 색: black
이름: Bella, 나이: 2, 색: white
```

```
# 문제4
# 원을 나타내는 클래스 Circle 생성
# Circle 클래스에 특별 메소드 추가하여 >와 < 그리고 == 연산자를 비교할 수 있도록 결정
class Circle:
  def __init__(self, radius):
    self.radius = radius

  def __gt__(self, other):
    return self.radius > other.radius
  def __lt__(self, other):
    return self.radius < other.radius
  def __eq__(self, other):
   return self.radius == other.radius

c1 = Circle(10)
c2 = Circle(7)

print(f"원 #1 = 반지름 {c1.radius}")
print(f"원 #2 = 반지름 {c2.radius}")
print(f"원 #1 > 원 #2는 {c1>c2}입니다.")
print(f"원 #1 < 원 #2는 {c1<c2}입니다.")
print(f"원 #1 == 원 #2는 {c1==c2}입니다.")
```

```
원 #1 = 반지름 10
원 #2 = 반지름 7
원 #1 > 원 #2는 True입니다.
원 #1 < 원 #2는 False입니다.
원 #1 == 원 #2는 False입니다.
```

```
# 문제5
# Person 클래스 정의 (이름, 나이)
# 객체의 정보를 출력하는 show()메소드 정의

class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age
  def show(self):
    print(f"이 사람의 이름은 {self.name}이고 나이는 {self.age}입니다.")

# Person 클래스를 상속받아서 Employee 클래스 정의 (연봉)
```

```
# show() 메소드를 재정의하여 연봉도 추가로 출력
class Employee(Person):
  def __init__(self, name, age, salary):
    super().__init__(name, age)
    self.salary = salary
  def show(self):
    print(f"이 사람의 이름은 {self.name}이고 나이는 {self.age}이고 연봉은 {self.salary}만원 입니다.")


p1 = Employee("Kim", 27, 5400)
p1.show()
```

```
이 사람의 이름은 Kim이고 나이는 27이고 연봉은 5400만원 입니다.
```

```
# 문제6
# 일반적인 사람을 나타내는 Person 클래스를 정의(이름, 나이)
# Hello를 출력하는 비공개 메서드 정의
# 비공개 메서드 정의하는 메서드 정의

class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def __Hello(self):
    print(f"안녕하세요 {self.name}님 좋은하루 보내세요")

  def greet(self):
    self.__Hello()

p1 = Person("Kim", 25)
p1.greet()
```

```
안녕하세요 Kim님 좋은하루 보내세요
```

```
# 문제7
# 사용자의 아이디와 패스워드를 저장하는 리스트 생성 (Account 클래스)
# 각 입력값에 맞는 함수 만들기 (아이디 등록, 로그인, 모든 사용자 출력, 종료)

class Account:
  def __init__(self, id, pw):
    self.id = id
    self.pw = pw

accounts = []

def register():
  id = input("아이디를 입력하세요: ")
  pw = input("패스워드를 입력하세요: ")
  accounts.append(Account(id, pw))

def login():
  l_id = input("아이디를 입력하세요: ")
  l_pw = input("비밀번호를 입력하세요: ")
  for ac in accounts:
    if ac.id == l_id and ac.pw == l_pw:
      print("로그인 성공")
      return
  print("로그인 실패")

def get_all_user():
  trial = 0
  for ac in accounts:
    print(f"{trial + 1}번째 사용자의 아이디는 {ac.id}, 비밀번호는 {ac.pw}")
    trial += 1

# while 사용하여 계속해서 입력값 얻기
action = True
while action:
  choice = int(input("""
  ==================================================
  1. 아이디 등록하기
  2. 로그인하기
  3. 모든 사용자 아이디 출력
  4. 종료
  ==================================================
  번호를 입력하시오: """))

  if choice == 1:
    # 아이디 등록 함수
    register()
```

```
    if choice == 2:
      # 로그인 함수
      login()

    if choice == 3:
      # 모든 사용자 아이디 출력
      get_all_user()
    if choice == 4:
      action = False
```

```
      ==================================================
      1. 아이디 등록하기
      2. 로그인하기
      3. 모든 사용자 아이디 출력
      4. 종료
      ==================================================
      번호를 입력하시오: 1
    아이디를 입력하세요: miinsu001
    패스워드를 입력하세요: jaesu002

      ==================================================
      1. 아이디 등록하기
      2. 로그인하기
      3. 모든 사용자 아이디 출력
      4. 종료
      ==================================================
      번호를 입력하시오: 1
    아이디를 입력하세요: abc
    패스워드를 입력하세요: ddc

      ==================================================
      1. 아이디 등록하기
      2. 로그인하기
      3. 모든 사용자 아이디 출력
      4. 종료
      ==================================================
      번호를 입력하시오: 2
    아이디를 입력하세요: abc
    비밀번호를 입력하세요: ddc
    로그인 성공

      ==================================================
      1. 아이디 등록하기
      2. 로그인하기
      3. 모든 사용자 아이디 출력
      4. 종료
      ==================================================
      번호를 입력하시오: 3
    1번째 사용자의 아이디는 miinsu001, 비밀번호는 jaesu002
    2번째 사용자의 아이디는 abc, 비밀번호는 ddc

      ==================================================
      1. 아이디 등록하기
      2. 로그인하기
      3. 모든 사용자 아이디 출력
      4. 종료
      ==================================================
      번호를 입력하시오: 4
```

```
# 문제8
# 정적 메소드 사용하기
class Calc:
  @staticmethod
  def add(a,b):
    print(a+b)

  @staticmethod
  def mul(a,b):
    print(a*b)

Calc.add(10,20)
Calc.mul(10,20)
```

```
    30
    200
```

```
# 문제9
# 클래스 메소드 사용하기
class Person:
  # 클래스 속성
  count = 0

  def __init__(self):
    Person.count += 1
```

```
    @classmethod
    def print_count(cls):
      print(f"{cls.count}명의 인스턴스가 존재합니다.")

james = Person()
minsu = Person()

Person.print_count()
```

```
2명의 인스턴스가 존재합니다.
```