ICP2 REPORT

1-1)-Counter. Count (Class Variable):

This is a **class variable**, meaning it is shared across all instances of the Counter class.

-self. Count (Instance Variable):

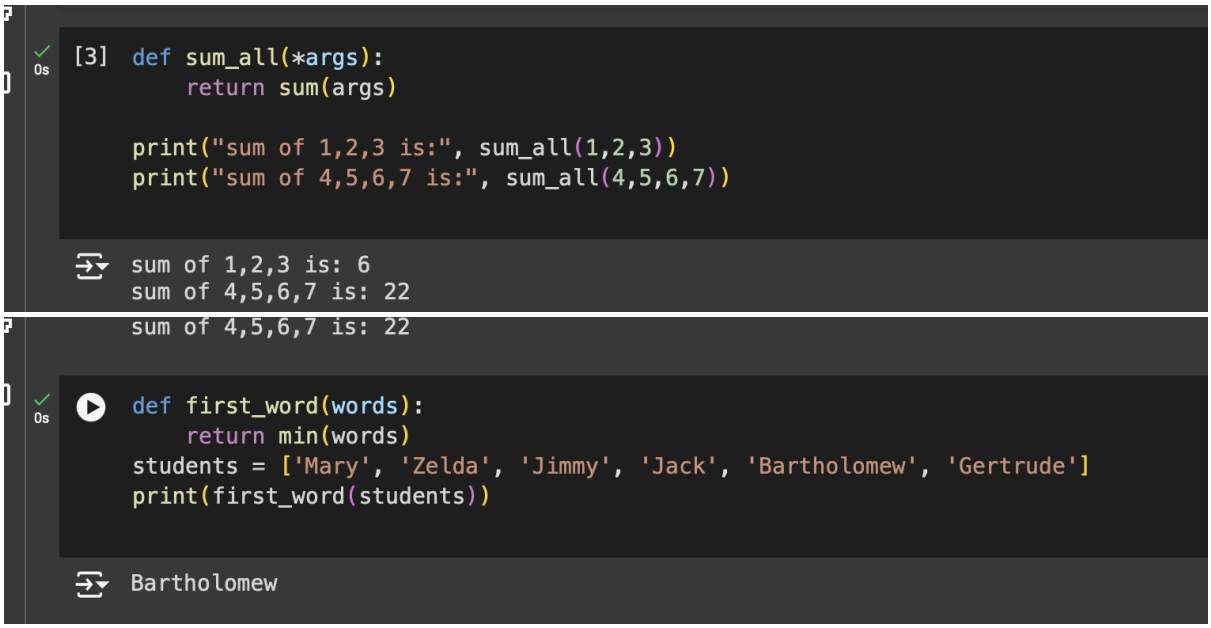
This is an **instance variable**, meaning it is unique to each instance of the Counter class.

```python
class Counter:
    count = 0

    def __init__(self):
        self._count = 0

    def increment(self):
        self._count += 1
        Counter.count += 1

    def get_counts(self):
        return f"Instance count: {self._count}, Class count: {Counter.count}"



a = Counter()
b = Counter()


a.increment()
a.increment()
b.increment()


print(a.get_counts())
print(b.get_counts())
```

```
Instance count: 2, Class count: 3
Instance count: 1, Class count: 3
```

1-3) The increment method in the Counter class affects both the class variable Counter.count and the instance variable self. Count

- **a.increment()**:
  - Increases a._count from 0 to 1.
  - Increases Counter.count from 0 to 1.
- **a.increment()**:
  - Increases a._count from 1 to 2.
  - Increases Counter.count from 1 to 2.
- **b.increment()**:
  - Increases b._count from 0 to 1.
  - Increases Counter.count from 2 to 3.



```
[3]  def sum_all(*args):
         return sum(args)

     print("sum of 1,2,3 is:", sum_all(1,2,3))
     print("sum of 4,5,6,7 is:", sum_all(4,5,6,7))
```

```
sum of 1,2,3 is: 6
sum of 4,5,6,7 is: 22
```

```
sum of 4,5,6,7 is: 22
```

```
def first_word(words):
    return min(words)
students = ['Mary', 'Zelda', 'Jimmy', 'Jack', 'Bartholomew', 'Gertrude']
print(first_word(students))
```

```
Bartholomew
```

```python
class Employee:

    employee_count = 0

    def __init__(self, name):
        self.name = name
        Employee.employee_count += 1


emp1 = Employee("Nithin")
emp2 = Employee("Aman")
emp3 = Employee("Nitish")

print("Total number of Employees:", Employee.employee_count)
```

```
Total number of Employees: 3
```

```python
[19] class Employee:
        employee_count = 0
        def __init__(self, name, family, salary, department):
            self.name = name
            self.family = family
            self.salary = salary
            self.department = department
            Employee.employee_count += 1
    emp1 = Employee("Nithin", "Reddy", 50000, "HR")
    emp2 = Employee("Aman", "Goud", 60000, "Finance")
    print(f"Employee 1: {emp1.name}, {emp1.family}, {emp1.salary}, {emp1.department}"
    print(f"Employee 2: {emp2.name}, {emp2.family}, {emp2.salary}, {emp2.department}"
    print("Total number of Employees:", Employee.employee_count)
```

```
Employee 1: Nithin, Reddy, 50000, HR
Employee 2: Aman, Goud, 60000, Finance
Total number of Employees: 2
```

```python
class Employee:

    employee_count = 0
    total_salary = 0

    def __init__(self, name,salary):
        self.name = name
        self.salary = salary
        Employee.employee_count += 1
        Employee.total_salary += salary

    @classmethod
    def average_salary(cls):
        if cls.employee_count == 0:
            return 0
        return cls.total_salary / cls.employee_count


emp1 = Employee("nithin", 50000)
emp2 = Employee("aman", 60000)
emp3 = Employee("nick",70000)

print("Average salary of employees:", Employee.average_salary())
```

```
Average salary of employees: 60000.0
```

```python
class Employee:

    employee_count = 0
    total_salary = 0

    def __init__(self, name,salary,):
        self.name = name
        self.salary = salary
        Employee.employee_count += 1
        Employee.total_salary += salary

    @classmethod
    def average_salary(cls):
        if cls.employee_count == 0:
            return 0
        return cls.total_salary / cls.employee_count


class FulltimeEmployee(Employee):
    def __init__(self, name, salary,benefits):
        super().__init__(name ,salary,)
        self.benefits = benefits


ft_emp1 = FulltimeEmployee("Nithin", 70000, ["Health Insurance"])
ft_emp2 = FulltimeEmployee("Aman", 80000, ["Health Insurance"])

print(f"Fulltime Employee 1: {ft_emp1.name}, {ft_emp1.salary},{ft_emp1.benefits}"
print(f"Fulltime Employee 2: {ft_emp2.name},  {ft_emp2.salary},  {ft_emp2.benefit
print("Average salary of employees:", Employee.average_salary())
```

```
Fulltime Employee 1: Nithin, 70000,['Health Insurance']
Fulltime Employee 2: Aman,  80000,  ['Health Insurance']
Average salary of employees: 75000.0
```

```python
class Employee:

    employee_count = 0
    total_salary = 0

    def __init__(self, name, salary,):
        self.name = name
        self.salary = salary
        Employee.employee_count += 1
        Employee.total_salary += salary

    @classmethod
    def average_salary(cls):
        if cls.employee_count == 0:
            return 0
        return cls.total_salary / cls.employee_count

    def display_info(self):
        return f"Name: {self.name},  Salary: {self.salary},"


class FulltimeEmployee(Employee):
    def __init__(self, name, salary,  benefits):

        super().__init__(name, salary,)
        self.benefits = benefits

    def display_info(self):
        employee_info = super().display_info()
        return f"{employee_info}, Benefits: {self.benefits}"


emp1 = Employee("akshay", 50000, )
emp2 = Employee("praveen", 60000, )
```

```python
emp2 = Employee("praveen", 60000, )


ft_emp1 = FulltimeEmployee("nithin", 70000, ["Health Insurance", ])
ft_emp2 = FulltimeEmployee("aman", 80000, ["Health Insurance",])


print(emp1.display_info())
print(emp2.display_info())
print(ft_emp1.display_info())
print(ft_emp2.display_info())
print("Average salary of all employees:", Employee.average_salary())
```

```
Name: akshay,  Salary: 50000,
Name: praveen,  Salary: 60000,
Name: nithin,  Salary: 70000,, Benefits: ['Health Insurance']
Name: aman,  Salary: 80000,, Benefits: ['Health Insurance']
Average salary of all employees: 65000.0
```

MY GITHUB LINK:https://github.com/nithin1086/BDA.git