1. Create a class Student with following

a. data members :

- **i. StudentId**
- **ii. Name**
- **iii. Age**
- **iv. Percentage**

b. Add the following methods :

- **i. Parameterized constructor**
- **ii. Display**
- **iii. Accept**
- **iv. Method CalculateRank**
- **v. Override __str__ Method**

```
class Student:
  #Parameterized constructor
  def __init__(self, student_id, name, age, percentage):
    self.student_id = student_id
    self.name = name
    self.age = age
    self.percentage = percentage

  #Accept method to take input from user
  def accept(self):
    self.student_id = int(input("Enter Student ID: "))
    self.name = input("Enter Name: ")
    self.age = int(input("Enter Age: "))
    self.percentage = float(input("Enter Percentage: "))

  #Display method
  def display(self):
    print(f"Student ID: {self.student_id}")
    print(f"Name: {self.name}")
    print(f"Age: {self.age}")
    print(f"Percentage: {self.percentage}%")
    print(f"Rank: {self.calculate_rank()}")

  #Method to calculate rank based on percentage
  def calculate_rank(self):
    if self.percentage >= 90:
      return "A+"
    elif self.percentage >= 75:
      return "A"
```

```
        elif self.percentage >= 60:
            return "B"
        elif self.percentage >= 40:
            return "C"
        else:
            return "Fail"

    #Override __str__ method
    def __str__(self):
        return (f"[Student ID: {self.student_id}, Name: {self.name}, "
                f"Age: {self.age}, Percentage: {self.percentage}%, "
                f"Rank: {self.calculate_rank()}]")

#Example usage
s1 = Student(101, "Alice", 20, 87.5)
s1.display()
print(s1)

print("\nCreate a new student using accept() method:")
s2 = Student(0, "", 0, 0.0)
s2.accept()
print(s2)
```

2. Create a derived class from Student as EnggStudent with :

a. Data members as :

- **i. Branch**
- **ii. InternalMarks**

b. Add the following methods :

- **i. Parameterized constructor**
- **ii. Display**
- **iii. Accept**
- **iv. override Method CalculateRank**
- **v. Override __str__ Method**

```
#Base class
class Student:
    def __init__(self, student_id, name, age, percentage):
        self.student_id = student_id
        self.name = name
        self.age = age
        self.percentage = percentage
```

```
  def accept(self):
    self.student_id = int(input("Enter Student ID: "))
    self.name = input("Enter Name: ")
    self.age = int(input("Enter Age: "))
    self.percentage = float(input("Enter Percentage: "))

  def display(self):
    print(f"Student ID: {self.student_id}")
    print(f"Name: {self.name}")
    print(f"Age: {self.age}")
    print(f"Percentage: {self.percentage}%")
    print(f"Rank: {self.calculate_rank()}")

  def calculate_rank(self):
    if self.percentage >= 90:
      return "A+"
    elif self.percentage >= 75:
      return "A"
    elif self.percentage >= 60:
      return "B"
    elif self.percentage >= 40:
      return "C"
    else:
      return "Fail"

  def __str__(self):
    return (f"[Student ID: {self.student_id}, Name: {self.name}, Age: {self.age}, "
        f"Percentage: {self.percentage}%, Rank: {self.calculate_rank()}]")

# Derived class
class EnggStudent(Student):
  def __init__(self, student_id, name, age, percentage, branch, internal_marks):
    super().__init__(student_id, name, age, percentage)
    self.branch = branch
    self.internal_marks = internal_marks
  def accept(self):
    super().accept()
    self.branch = input("Enter Branch: ")
    self.internal_marks = float(input("Enter Internal Marks: "))

  def display(self):
    super().display()
    print(f"Branch: {self.branch}")
    print(f"Internal Marks: {self.internal_marks}")

  def calculate_rank(self):
    final_score = (self.percentage + self.internal_marks) / 2
    if final_score >= 90:
      return "A+"
    elif final_score >= 75:
      return "A"
    elif final_score >= 60:
      return "B"
    elif final_score >= 40:
      return "C"
    else:
      return "Fail"

  def __str__(self):
    return (f"[EnggStudent ID: {self.student_id}, Name: {self.name}, Age: {self.age}, "
        f"Percentage: {self.percentage}%, Branch: {self.branch}, "
        f"Internal Marks: {self.internal_marks}, Rank: {self.calculate_rank()}]")

#Example usage
e1 = EnggStudent(201, "Bob", 21, 82.0, "Computer", 88.0)
e1.display()
print("\nUsing __str__ override:", e1)

print("\nCreate a new engineering student using accept() method:")
e2 = EnggStudent(0, "", 0, 0.0, "", 0.0)
e2.accept()
print(e2)
```

3. Create a class MedicalStudent inherited from Student with following:

i. Data members :Specialization

ii. MarksOfInternship

b. Add the following methods :

i. Parameterized constructor

ii. Display

iii. Accept

iv. override Method CalculateRank

v. Override __str__ Method

```
class Student:
  def __init__(self, student_id, name, age, percentage):
    self.student_id = student_id
    self.name = name
    self.age = age
    self.percentage = percentage

  def accept(self):
    self.student_id = int(input("Enter Student ID: "))
    self.name = input("Enter Name: ")
    self.age = int(input("Enter Age: "))
    self.percentage = float(input("Enter Percentage: "))

  def display(self):
    print(f"Student ID: {self.student_id}")
    print(f"Name: {self.name}")
    print(f"Age: {self.age}")
    print(f"Percentage: {self.percentage}%")
    print(f"Rank: {self.calculate_rank()}")

  def calculate_rank(self):
    if self.percentage >= 90:
      return "A+"
    elif self.percentage >= 75:
      return "A"
    elif self.percentage >= 60:
      return "B"
    elif self.percentage >= 40:
      return "C"
    else:
      return "Fail"
    def __str__(self):
        return (f"[Student ID: {self.student_id}, Name: {self.name}, Age: {self.age}, "
                f"Percentage: {self.percentage}%, Rank: {self.calculate_rank()}]")

# Derived class
class MedicalStudent(Student):
    def __init__(self, student_id, name, age, percentage, specialization,
marks_of_internship):
        super().__init__(student_id, name, age, percentage)
        self.specialization = specialization
        self.marks_of_internship = marks_of_internship

    def accept(self):
        super().accept()
        self.specialization = input("Enter Specialization: ")
        self.marks_of_internship = float(input("Enter Internship Marks: "))

    def display(self):
        super().display()
        print(f"Specialization: {self.specialization}")
        print(f"Internship Marks: {self.marks_of_internship}")

    def calculate_rank(self):
        final_score = (self.percentage + self.marks_of_internship) / 2
        if final_score >= 90:
            return "A+"
        elif final_score >= 75:
            return "A"
        elif final_score >= 60:
            return "B"
        elif final_score >= 40:
            return "C"
        else:
            return "Fail"
    def __str__(self):
        return (f"[MedicalStudent ID: {self.student_id}, Name: {self.name}, Age: 
{self.age}, "
                f"Percentage: {self.percentage}%, Specialization: {self.specialization}, "
                f"Internship Marks: {self.marks_of_internship}, Rank: 
{self.calculate_rank()}]")


m1 = MedicalStudent(301, "Dr. Sara", 23, 85.0, "Cardiology", 90.0)
m1.display()
print("\nUsing __str__ override:", m1)


print("\nCreate a new medical student using accept() method:")
m2 = MedicalStudent(0, "", 0, 0.0, "", 0.0)
m2.accept()
print(m2)
```