In-Lab

In-Lab Task 1

Code:

```
def countEvenNumbers(numberList):
    evenCount = 0
    for number in numberList:
        if (number % 2 == 0):
            evenCount += 1
    print(evenCount)
#Example lists
numberList1 = [1,2,3,4,5,6,7,8,9,10]
numberList2 = [11,13,15,17]
numberList3 = []
#Calling the function
countEvenNumbers(numberList1)
countEvenNumbers(numberList2)
countEvenNumbers(numberList3)
```

Output:

```
5
0
0
```

In-Lab Task 2

Code:

```
def calculateGPA(studentData):
    result = []

    for student in studentData:
        name = student['name']
        marks = student['marks']
        totalGradePoints = 0
        totalCourses = len(marks)
        grades = []
#Rather than using Hash-Map, I used if and elif statements.
        for mark in marks:
            if 85 <= mark <= 100:
                grade = 'A'
                gradePoint = 4.00
            elif 80 <= mark <= 84:
                grade = 'A-'
                gradePoint = 3.66
```

```
            elif 75 <= mark <= 79:
                grade = 'B+'
                gradePoint = 3.33
            elif 71 <= mark <= 74:
                grade = 'B'
                gradePoint = 3.00
            elif 68 <= mark <= 70:
                grade = 'B-'
                gradePoint = 2.66
            elif 64 <= mark <= 67:
                grade = 'C+'
                gradePoint = 2.33
            elif 61 <= mark <= 63:
                grade = 'C'
                gradePoint = 2.00
            elif 58 <= mark <= 60:
                grade = 'C-'
                gradePoint = 1.66
            elif 54 <= mark <= 57:
                grade = 'D+'
                gradePoint = 1.30
            elif 50 <= mark <= 53:
                grade = 'D'
                gradePoint = 1.00
            else:
                grade = 'F'
                gradePoint = 0.00

            grades.append(grade)
            totalGradePoints += gradePoint

        gpa = totalGradePoints / totalCourses if totalCourses > 0
else 0.00

        studentInfo = {
            'name': name,
            'grades': grades,
            'gradePoints': totalGradePoints,
            'gpa': round(gpa, 2)
        }

        result.append(studentInfo)

    return result
```

```
# Data for five students
students = [
    {'name': 'Rana Fahad Aman', 'marks': [85, 75, 92, 68, 60]},
    {'name': 'Afaan Kamran', 'marks': [78, 88, 70, 92, 81]},
    {'name': 'Shaheer Farhan', 'marks': [62, 53, 45, 75, 80]},
    {'name': 'Malaika Asghar', 'marks': [95, 89, 92, 78, 86]},
    {'name': 'Daud Hassan', 'marks': [45, 55, 50, 62, 75]},
]

# Calculating GPA of the students
gpaResults = calculateGPA(students)

# Printing the results
for studentInfo in gpaResults:
    print(f"Name: {studentInfo['name']}")
    print(f"Grades: {', '.join(studentInfo['grades'])}")
    print(f"Grade Points: {studentInfo['gradePoints']}")
    print(f"GPA: {studentInfo['gpa']}")
```

Output:

```
Name: Rana Fahad Aman
Grades: A, B+, A, B-, C-
Grade Points: 15.65
GPA: 3.13

Name: Afaan Kamran
Grades: B+, A, B-, A, A-
Grade Points: 17.65
GPA: 3.53

Name: Shaheer Farhan
Grades: C, D, F, B+, A-
Grade Points: 9.99
GPA: 2.0

Name: Malaika Asghar
Grades: A, A, A, B+, A
Grade Points: 19.33
GPA: 3.87

Name: Daud Hassan
Grades: F, D+, D, C, B+
Grade Points: 7.63
GPA: 1.53
```

In-Lab Task 3

Code:

```
class Student:
    def __init__(self, name, rollNumber, *marks):
        self.name = name
        self.rollNumber = rollNumber
        self.marks = list(marks)  # Convert marks to a list
        print(f"Marks at Object Initialiation: {self.marks}")

    def addMarks(self, mark):
        self.marks.append(mark)
        print(f"Updated marks after addition: {self.marks}")

    def calculateAverage(self):
        totalMarks = sum(self.marks)
        totalMarksEntries = len(self.marks)
        if totalMarksEntries > 0:
            print(f"The average marks of {self.name} is {totalMarks /
totalMarksEntries:.2f}")
        else:
            print("No marks available for calculation.")

# Create an instance of the Student class
student1 = Student("Rana Fahad Aman", 21, 40, 50, 60)

print(f"Student Name: {student1.name}")
print(f"Roll Number: {student1.rollNumber}")
print(f"Marks: {student1.marks}")

# Add more marks
student1.addMarks(70)
student1.addMarks(55)
# Calculate the average marks
student1.calculateAverage()
```

Output:

```
Marks at Object Initialiation: [40, 50, 60]
Student Name: Rana Fahad Aman
Roll Number: 21
Marks: [40, 50, 60]
Updated marks after addition: [40, 50, 60, 70]
Updated marks after addition: [40, 50, 60, 70, 55]
The average marks of Rana Fahad Aman is 55.00
```

Post-Lab

Post-Lab Task

Code:

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
        self.available = True
    def borrow(self):
        if self.available:
            self.available = False
            return f"You have borrowed '{self.title}' by
{self.author}."
        else:
            return f"'{self.title}' is currently not available as it
already has been borrowed by someone."
    def returnBook(self):
        if not self.available:
            self.available = True
            return f"You have returned '{self.title}' by
{self.author}. Thank you!"
        else:
            return f"'{self.title}' by {self.author} has already been
returned."
book1 = Book("Charlie and the Chocolate Factory", "Roald Dahl")
book2 = Book("To Kill a Mockingbird", "Harper Lee")
book3 = Book("Harry Potter", "J.K. Rowling")
print(book1.borrow())
print(book2.borrow())
print(book1.returnBook())
print(book2.returnBook())
print(book2.returnBook())
print(book3.borrow())
print(book3.borrow())
```

Output:

```
You have borrowed 'Charlie and the Chocolate Factory' by Roald Dahl.
You have borrowed 'To Kill a Mockingbird' by Harper Lee.
You have returned 'Charlie and the Chocolate Factory' by Roald Dahl. Thank you!
You have returned 'To Kill a Mockingbird' by Harper Lee. Thank you!
'To Kill a Mockingbird' by Harper Lee has already been returned.
You have borrowed 'Harry Potter' by J.K. Rowling.
'Harry Potter' is currently not available as it already has been borrowed by someone.
```