Python Lab Exercise

1. Write a program display “Hello World!

```
print("Hello World")
```

```
Hello World
```

2. Write a program to find sum of two numbers. Use input() function to take input from the user.

```
a = int(input("a = "))
b = int(input("b = "))
sum  = a + b
print("Sum is:", sum)
```

```
a = 5
b = 3
Sum is: 8
```

3. Write a program to calculate discount on the basis of following assumption:

a) If purchased amount is greater than or equal to 1000, discount is 5%

```
amount  = int(input('Enter Purchased Amount'))
discount = 0
if amount >=1000:
  discount = amount * 0.05
print("Discount is:", discount)
```

```
Enter Purchased Ammount2000
Discount is: 100.0
```

4. Write a program to calculate discount on the basis of following assumption:

a) If purchased amount is greater than or equal to 1000, discount is 5% b) If purchased amount is less than 1000, discount is 3%

```
amount  = int(input('Enter Purchased Amount'))
discount = 0
if amount >=1000:
  discount = amount * 0.05
else:
  discount = amount * 0.03
print(f"Discount is:{discount}")
```

```
Enter Purchased Ammount500
Discount is:15.0
```

5. Write a program to calculate discount on the basis of following assumption:

- a) If purchased amount is greater than or equal to 5000, discount is 10%
- b) If purchased amount is greater than or equal to 4000 and less than 5000, discount is 7%
- c) If purchased amount is greater than or equal to 3000 and less than 4000, discount is 5%
- d) If purchased amount is greater than or equal to 2000 and less than 3000, discount is 3%
- e) If purchased amount is less than 2000, discount is 2%

```
amount  = int(input('Enter Purchased Amount'))
discount = 0
if amount >=5000:
  discount = amount * 0.1
elif amount >=4000:
  discount = amount * 0.07
elif amount >= 3000:
  discount = amount * 0.05
elif amount >= 2000:
```

```
  discount = amount * 0.03
else:
  discount = amount * 0.02
print(f"Discount is:{discount}")
```

```
Enter Purchased Amount3000
Discount is:150.0
```

6. Write a program to calculate the simple interest on the basis of following assumption:

- a) If balance is greater than 99999, interest is 7 %
- b) If balance is greater than or equal to 50000 and less than 100000 interest is 5 %
- c) If balance is less than 50000, interest is 3%

```
balance = float(input("Enter Your Balance:"))
interest = 0
if balance >= 99999:
  interest = balance * 0.07
elif (balance >= 50000 or balance <= 100000):
  interest = balance * 0.05
else:
  interest = balance * 0.03
print(f"Interest is:{interest}")
```

```
Enter Your Balance:50001
Interest is:2500.05
```

7 Write a program to test whether a number is even or odd.

```
number = int(input("Enter a Number:"))
# Check if the number is even or odd
if number % 2 == 0:
    print(f"{number} is even.")
else:
    print(f"{number} is odd.")
```

```
Enter a Number:6
6 is even.
```

8 Admission to a professional course is subject to the following conditions:

- a) Marks in mathematics >=60
- b) Marks in physics >=50
- c) Marks in chemistry >=40
- d) Total in all three subjects >=200 Or
- Total in mathematics and physics>=150

Given the marks in three subjects, write a program to process the applications to list eligible candidates.

```
# Input marks in mathematics, physics, and chemistry from the user
math_marks = int(input("Enter marks in Mathematics: "))
physics_marks = int(input("Enter marks in Physics: "))
chemistry_marks = int(input("Enter marks in Chemistry: "))

# Calculate the total marks in all three subjects
total_marks = math_marks + physics_marks + chemistry_marks

# Check if the conditions for admission are met
if (
    math_marks >= 60 and
    physics_marks >= 50 and
    chemistry_marks >= 40 and
    total_marks >= 200
) or (
    total_marks >= 150 and math_marks >= 60 and physics_marks >= 50
):
    print("Congratulations! You are eligible for admission to the professional course.")
else:
    print("Sorry, you are not eligible for admission to the professional course.")
```

```
Enter marks in Mathematics: 60
Enter marks in Physics: 50
Enter marks in Chemistry: 50
Congratulations! You are eligible for admission to the professional course.
```

9. The rates of tax on gross salary are as shown below:

Income	Tax
Less than 10,000	Nill
Rs. 10,000 to 19,999	10%
Rs. 20,000 to 39,999	15%
Rs. 40,000 to above	20%

Write a program to compute the net salary after deducting the tax for the given information and print the same.

```
# Input the gross salary from the user
gross_salary = float(input("Enter gross salary: "))

# Calculate the tax based on the provided income tax rates
if gross_salary < 10000:
    tax = 0
elif gross_salary < 20000:
    tax = 0.10 * gross_salary
elif gross_salary < 40000:
    tax = 0.15 * gross_salary
else:
    tax = 0.20 * gross_salary

# Calculate the net salary
net_salary = gross_salary - tax

# Display the results
print(f"Gross Salary: Rs. {gross_salary:.2f}")
print(f"Income Tax: Rs. {tax:.2f}")
print(f"Net Salary: Rs. {net_salary:.2f}")
```

```
Enter gross salary: 9999
Gross Salary: Rs. 9999.00
Income Tax: Rs. 0.00
Net Salary: Rs. 9999.00
```

10. Write a program to check whether a year entered is leap or not.

```
# Input a year from the user
y = int(input("Enter a year: "))

# Check if it's a leap year
if (y % 4 == 0) and ((y % 400 == 0) or (y % 100!=0)):
```

```
    print(f"{y} is a leap year")
else:
    print(f"{y} is not a leap year")
```

```
 Enter a year: 2024
 2024 is a leap year
```

11. write a program to display “MDS” 10 times

```
# Using a for loop to display "MDS" 10 times
for i in range(10):
    print("MDS")

# # Initialize a counter
# counter = 0

# # Use a while loop to display "MDS" 10 times
# while counter < 10:
#     print("MDS")
#     counter += 1
```

```
 MDS
 MDS
 MDS
 MDS
 MDS
 MDS
 MDS
 MDS
 MDS
 MDS
```

12. Write a program to find sum and average of 10 numbers stored in a list.

```
# Initialize an empty list to store the numbers
numbers = []

# Input 10 numbers from the user and store them in the list
for i in range(10):
    num = float(input(f"Enter number {i + 1}: "))
    numbers.append(num)

# Calculate the sum of the numbers
total_sum = 0
for num in numbers:
    total_sum += num

# Calculate the average
average = total_sum / len(numbers)

# Display the sum and average
print(f"Sum of the numbers: {total_sum}")
print(f"Average of the numbers: {average}")
```

```
##########################333
# numbers = [1,2,3,4,5,6,7,8,9,10]
# add = 0
# length = len(numbers)
# for i in numbers:
#     add = add + i
#     average = add/ length
# print(add)
# print(average)
```

```
 Enter number 1: 1
 Enter number 2: 2
 Enter number 3: 3
 Enter number 4: 4
 Enter number 5: 5
 Enter number 6: 6
```

```
Enter number 7: 7
Enter number 8: 8
Enter number 9: 9
Enter number 10: 10
Sum of the numbers: 55.0
Average of the numbers: 5.5
```

13. Write a program to display prime numbers up to 100.

```
# Loop through numbers from 2 to 100
for x in range(2, 101):
    is_prime = True  # Assume x is prime initially
    for y in range(2, int(x/2) + 1):
        if x % y == 0:
            is_prime = False  # If divisible, set is_prime to False
            break
    if is_prime:
        print(x, end=" ")
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

14. Write a program to find sum of digits of a number.

```
# Input a number from the user
number = int(input("Enter a number: "))

sum_of_digits = 0 # initialize variable
while number > 0:
  digit = number % 10 #get the last digit
  sum_of_digits += digit #add the digit to the sum
  number //= 10 #remove the last digit


# Display the sum of digits
print(f"The sum of digits is: {sum_of_digits}")
```

```
Enter a number: 2345
The sum of digits is: 14
```

15. Write a program to check whether a number is palindrome or not

```
# Input a number from the user
number = int(input("Enter a number: "))

# Convert the number to a string
number_str = str(number)

# Reverse the string
reverse_str = number_str[::-1]

# Check if the original number is equal to its reverse
if number_str == reverse_str:
    print(f"{number} is a palindrome.")
else:
    print(f"{number} is not a palindrome.")
```

```
Enter a number: 1211
1211 is not a palindrome.
```

```
# Input a number from the user
number = int(input("Enter a number: "))

# Make a copy of the original number
original_number = number

# Initialize a variable to store the reversed number
reversed_number = 0
```

```
# Reverse the digits of the number
while number > 0:
    digit = number % 10
    reversed_number = reversed_number * 10 + digit
    number //= 10

# Check if the reversed number is equal to the original number
if original_number == reversed_number:
    print(f"{original_number} is a palindrome.")
else:
    print(f"{original_number} is not a palindrome.")
```

```
Enter a number: 121
121 is a palindrome.
```

16. Write a program to check if a number is Armstrong Number or not

An Armstrong number is a number that is equal to the sum of its own digits each raised to the power of the number of digits.

```
# Input a number from the user
number = int(input("Enter a number: "))

# Count the number of digits in the number
num_digits = len(str(number))

# Initialize a variable to store the sum of digits raised to the power of num_digits
sum_of_digits = 0
temp = number

# Calculate the sum of digits raised to the power of num_digits
while temp > 0:
    digit = temp % 10
    sum_of_digits += digit ** num_digits
    temp //= 10

# Check if the number is an Armstrong number
if number == sum_of_digits:
    print(f"{number} is an Armstrong number.")
else:
    print(f"{number} is not an Armstrong number.")
```

```
Enter a number: 153
153 is an Armstrong number.
```

17. Write a program to count number of vowels in a string

```
# words = input("Please Enter your words:")
# count=0
# for i in words:
#       if(i=='a' or i=='e' or i=='i' or i=='o' or i=='u' or i=='A' or i=='E' or i=='I' or i=='O' or i=='U'):
#             count=count+1
# print("Number of vomels are:", count)

# Input a string from the user
string = input("Enter a string: ")
vowel_count = 0
vowels = set("AEIOUaeiou")
# Iterate through the characters in the string
for char in string:
    if char in vowels:
        vowel_count += 1

# Display the count of vowels
print(f"Number of vowels in the string: {vowel_count}")
```

```
Enter a string: hello world
Number of vowels in the string: 3
```

18. Write a program to find smallest and largest number among 10 numbers stored in a list

```
# Initialize an empty list to store the numbers
numbers = []

# Input 10 numbers from the user and store them in the list
for i in range(10):
    num = float(input(f"Enter number {i + 1}: "))
    numbers.append(num)

# Initialize variables for the smallest and largest numbers
smallest = numbers[0]
largest = numbers[0]

# Iterate through the list to find the smallest and largest numbers
for num in numbers:
    if num < smallest:
        smallest = num
    elif num > largest:
        largest = num

# Display the smallest and largest numbers
print(f"Smallest number: {smallest}")
print(f"Largest number: {largest}")

#####################################3
# # Input 10 numbers from the user and store them in a list
# numbers = [float(input(f"Enter number {i + 1}: ")) for i in range(10)]

# # Find the smallest and largest numbers using min() and max() functions
# smallest = min(numbers)
# largest = max(numbers)

# # Display the smallest and largest numbers
# print(f"Smallest number: {smallest}")
# print(f"Largest number: {largest}")
```

```
Enter number 1: 24
Enter number 2: 36
Enter number 3: 22
Enter number 4: 17
Enter number 5: 18
Enter number 6: 6
Enter number 7: 93
Enter number 8: 55
Enter number 9: 77
Enter number 10: 20
Smallest number: 6.0
Largest number: 93.0
```

```
#this can be acheived simply as well
numbers = [34,1,8,35,69,427,56,78]
sort_number = sorted(numbers)
print("largest_number", sort_number[-1])
print("smallest_number", sort_number[0])
```

```
largest_number 427
smallest_number 1
```

19. Write a program to count even numbers and odd numbers stored in a list

```
#numbers = [15,72,44,23,35,69,73,54,68]
#also can get the input as below
numbers = [float(input(f"Enter number {i + 1}: ")) for i in range(10)]

even = 0
odd = 0
for n in numbers:
    if n %2 == 0:
        even +=1
    else:
        odd +=1
print("Count of Even Numbers in a Given list is: ", even)
```

```
print("Count of Odd Numbers in a Given list is: ", odd)
```

```
Enter number 1: 23
Enter number 2: 64
Enter number 3: 35
Enter number 4: 22
Enter number 5: 91
Enter number 6: 3
Enter number 7: 4
Enter number 8: 5
Enter number 9: 77
Enter number 10: 65
Count of Even Numbers in a Given list is:  3
Count of Odd Numbers in a Given list is:  7
```

20. Write a program to find sum of two matrices.

```
a = [[1,2,3],[2,3,4]]
b = [[1,2,3],[4,5,7]]
c = [[0,0,0],[0,0,0]]
for i in range(2):
    for j in range(3):
        c[i][j] = a[i][j] + b[i][j]

for r in c:
    print(r)
```

```
[2, 4, 6]
[6, 8, 11]
```

```
#using numpy
import numpy as np

# Input the elements of the first matrix
matrix1 = np.array([[3.0, 4.0, 5.0], [1.0, 2.0, 3.0]])

# Input the elements of the second matrix
matrix2 = np.array([[2.0, 1.0, 3.0], [3.0, 5.0, 6.0]])

# Calculate the sum of the matrices
result_matrix = np.add(matrix1, matrix2)

# Display the result
print("Sum of the matrices:")
print(result_matrix)
```

```
Sum of the matrices:
[[5. 5. 8.]
 [4. 7. 9.]]
```

21. Write a program to find product of two matrices.

```
a = [[1, 2, 3], [2, 3, 4], [4, 5, 7]]
b = [[1, 2, 3], [4, 5, 7], [3, 4, 6]]
c = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]

# Perform matrix multiplication
for i in range(len(a)):
    for j in range(len(b[0])):
        for k in range(len(b)):
            c[i][j] += a[i][k] * b[k][j]

# Display the result matrix
print("Result of matrix multiplication:")
for row in c:
    print(row)
```

```
Result of matrix multiplication:
[18, 24, 35]
[26, 35, 51]
[45, 61, 89]
```

```
#using numpy array
import numpy as np

a = np.array([[1, 2, 3], [2, 3, 4], [4, 5, 7]])
b = np.array([[1, 2, 3], [4, 5, 7], [3, 4, 6]])

# Perform matrix multiplication using numpy.dot or @ operator
c = np.dot(a, b)

# Display the result matrix
print("Result of matrix multiplication:")
print(c)
```

```
Result of matrix multiplication:
[[18 24 35]
 [26 35 51]
 [45 61 89]]
```

22. Write a program using list comprehension to find sum of only even numbers.

```
# Input a list of numbers from the user
numbers = [5, 3, 2, 4, 8, 44, 6]

# Use list comprehension to create a list of even numbers
even_numbers = [num for num in numbers if num % 2 == 0]
odd_numbers = [num for num in numbers if num % 2 != 0]

# Calculate the sum of the even numbers
even_sum = 0
for num in even_numbers:
    even_sum += num

# Calculate the sum of the odd numbers
odd_sum = 0
for num in odd_numbers:
    odd_sum += num

print(f"Sum of even numbers: {even_sum}")
print(f"Sum of odd numbers: {odd_sum}")
```

```
Sum of even numbers: 64
Sum of odd numbers: 8
```

23. Write a program using function with return type to find sum of two numbers.

```
# Define a function to find the sum of two numbers
def add_numbers(num1, num2):
    result = num1 + num2
    return result

# Input two numbers from the user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

# Call the function to find the sum
sum_result = add_numbers(num1, num2)

# Display the result
print(f"The sum of {num1} and {num2} is: {sum_result}")
```

```
Enter the first number: 20
Enter the second number: 30
The sum of 20.0 and 30.0 is: 50.0
```

24. Write a program using recursive function to find factorial of a number.

```
# Define a recursive function to find factorial
def factorial(n):
    if n == 0:
        return 1  # Base case: 0! is defined as 1
    else:
        return n * factorial(n - 1)

# Input a number from the user
num = int(input("Enter a non-negative integer: "))

# Check if the input is non-negative
if num < 0:
    print("Factorial is not defined for negative numbers.")
else:
    result = factorial(num)
    print(f"The factorial of {num} is: {result}")
```

```
 Enter a non-negative integer: 5
 The factorial of 5 is: 120
```

▾ 25. Write a program using recursive function to find nth Fibonacci number

```
# Define a recursive function to find the nth Fibonacci number
def fibonacci(n):
    if n <= 0:
        return 0  # Base case: Fibonacci(0) = 0
    elif n == 1:
        return 1  # Base case: Fibonacci(1) = 1
    else:
        return fibonacci(n - 1) + fibonacci(n - 2)

# Input a positive integer from the user
n = int(input("Enter a positive integer (n): "))

# Check if the input is positive
if n < 0:
    print("Please enter a positive integer.")
else:
    result = fibonacci(n)
    print(f"The {n}th Fibonacci number is: {result}")
```

```
 Enter a positive integer (n): 17
 The 17th Fibonacci number is: 1597
```

▾ 26. Create a class Rectangle containing instance variables length and breadth. The class also contains two instance methods area() and perimeter() to find area and perimeter of rectangles respectively. Use this class to find area and perimeter of two different rectangles.

```
class Rectangle:
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

    def perimeter(self):
        return 2 * (self.length + self.breadth)

# Create two different rectangle objects
rectangle1 = Rectangle(5, 10)
rectangle2 = Rectangle(3, 7)

# Calculate and print the area and perimeter of each rectangle
print("Rectangle 1:")
print(f"Area: {rectangle1.area()}")
print(f"Perimeter: {rectangle1.perimeter()}\n")

print("Rectangle 2:")
```

```
print(f"Area: {rectangle2.area()}")
print(f"Perimeter: {rectangle2.perimeter()}")
```

```
Rectangle 1:
Area: 50
Perimeter: 30

Rectangle 2:
Area: 21
Perimeter: 20
```

27. Create a class Circle containing an instance variable radius. The class also contains two instance methods area() and circumference() to find area and circumference of circles respectively. Use this class to find area and circumference of two different circles. Use PI as a class variable

```
class Circle:
    # Class variable for PI
    PI = 3.14159265359

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

    def area(self):
        return Circle.PI * self.radius ** 2

    def circumference(self):
        return 2 * Circle.PI * self.radius

# Create two different circle objects
circle1 = Circle(5)
circle2 = Circle(3)

# Calculate and print the area and circumference of each circle
print("Circle 1:")
print(f"Area: {circle1.area()}")
print(f"Circumference: {circle1.circumference()}\n")

print("Circle 2:")
print(f"Area: {circle2.area()}")
print(f"Circumference: {circle2.circumference()}")
```

```
Circle 1:
Area: 78.53981633975
Circumference: 31.4159265359

Circle 2:
Area: 28.27433388231
Circumference: 18.849555921540002
```

28. Create a class Box with instance variables width, height and depth. The class also contains instance methods volume() and surface_area() to find volume and surface area of boxes respectively. Use this class to find volume and surface area of two different boxes

```
class Box:
    def __init__(self, width, height, depth):
        self.width = width
        self.height = height
        self.depth = depth

    def volume(self):
        return self.width * self.height * self.depth

    def surface_area(self):
        return 2 * (self.width * self.height + self.width * self.depth + self.height * self.depth)

# Create two different box objects
box1 = Box(5, 3, 4)
box2 = Box(2, 2, 2)
```

```
# Calculate and print the volume and surface area of each box
print("Box 1:")
print(f"Volume: {box1.volume()}")
print(f"Surface Area: {box1.surface_area()}\n")

print("Box 2:")
print(f"Volume: {box2.volume()}")
print(f"Surface Area: {box2.surface_area()}")
```

```
Box 1:
Volume: 60
Surface Area: 94

Box 2:
Volume: 8
Surface Area: 24
```

29. Create a class Time with three instance variables hours, minutes, and seconds. Add instance methods display() to display the time in hh:mm:ss format and add() to add two time objects. Use this class to add and display two different time objects

```
class Time:
    def __init__(self, hours, minutes, seconds):
        self.hours = hours
        self.minutes = minutes
        self.seconds = seconds

    def display(self):
        return f"{self.hours:02d}:{self.minutes:02d}:{self.seconds:02d}"

    def add(self, other_time):
        total_seconds = self.hours * 3600 + self.minutes * 60 + self.seconds
        total_seconds += other_time.hours * 3600 + other_time.minutes * 60 + other_time.seconds

        # Calculate the new time components
        new_hours, remainder = divmod(total_seconds, 3600)
        new_minutes, new_seconds = divmod(remainder, 60)

        return Time(new_hours, new_minutes, new_seconds)

# Create two different time objects
time1 = Time(4, 30, 15)
time2 = Time(2, 45, 30)

# Display the first time
print("Time 1:", time1.display())

# Display the second time
print("Time 2:", time2.display())

# Add the two time objects
sum_time = time1.add(time2)

# Display the sum of the times
print("Sum of Time 1 and Time 2:", sum_time.display())
```

```
Time 1: 04:30:15
Time 2: 02:45:30
Sum of Time 1 and Time 2: 07:15:45
```

30. Create a class Distance containing instance variables feet and inches. The class also contains instance methods add() and compare() to add and compare two distance objects respectively.

Use this class to create two different distance objects and add and compare these two distance object

```
class Distance:
    def __init__(self, feet, inches):
        self.feet = feet
        self.inches = inches
```

```
        def add(self, other_distance):
            total_feet = self.feet + other_distance.feet
            total_inches = self.inches + other_distance.inches

            if total_inches >= 12:
                total_feet += total_inches // 12
                total_inches %= 12

            return Distance(total_feet, total_inches)

        def compare(self, other_distance):
            if self.feet > other_distance.feet:
                return "First distance is greater"
            elif self.feet < other_distance.feet:
                return "Second distance is greater"
            else:
                if self.inches > other_distance.inches:
                    return "First distance is greater"
                elif self.inches < other_distance.inches:
                    return "Second distance is greater"
                else:
                    return "Both distances are equal"

        def display(self):
            return f"{self.feet} feet {self.inches} inches"

    # Create two different distance objects
    distance1 = Distance(5, 8)
    distance2 = Distance(4, 10)

    # Display the first distance
    print("Distance 1:", distance1.display())

    # Display the second distance
    print("Distance 2:", distance2.display())

    # Add the two distance objects
    sum_distance = distance1.add(distance2)

    # Display the sum of the distances
    print("Sum of Distance 1 and Distance 2:", sum_distance.display())

    # Compare the two distance objects
    comparison_result = distance1.compare(distance2)
    print("Comparison result:", comparison_result)
```

```
     Distance 1: 5 feet 8 inches
     Distance 2: 4 feet 10 inches
     Sum of Distance 1 and Distance 2: 10 feet 6 inches
     Comparison result: First distance is greater
```

31. Create a class Student with instance variables name, roll_number, and marks in five subjects.

- Add three instance methods in this class to calculate total(), percentage(), and division() of the marks obtained by the students.
- Use this class to find total marks obtained, percentage, and division of five students.

```
class Student:

    def __init__(self, name, roll_number, marks):
        self.name = name
        self.roll_number = roll_number
        self.marks = marks

    # def total(self):
    #     return sum(self.marks)

    def total(self):
      total_marks = 0
      for mark in self.marks:
        total_marks += mark
      return total_marks

    def percentage(self):
        total_marks = self.total()
        return (total_marks / (len(self.marks) * 100)) * 100
```

```
            return (total_marks / (len(self.marks) * 100)) * 100

        def division(self):
            percentage = self.percentage()
            if percentage >= 60:
                return "First Division"
            elif percentage >= 45:
                return "Second Division"
            elif percentage >= 33:
                return "Third Division"
            else:
                return "Fail"

    # Create instances for five students
    student1 = Student("Alice", "101", [75, 80, 85, 90, 95])
    student2 = Student("Bob", "102", [65, 70, 75, 80, 85])
    student3 = Student("Charlie", "103", [55, 60, 65, 70, 75])
    student4 = Student("David", "104", [45, 50, 55, 60, 65])
    student5 = Student("Eve", "105", [35, 40, 45, 50, 55])

    # Calculate and display total marks, percentage, and division for each student
    students = [student1, student2, student3, student4, student5]
    for student in students:
        print(f"Name: {student.name}")
        print(f"Roll Number: {student.roll_number}")
        print(f"Total Marks: {student.total()}")
        print(f"Percentage: {student.percentage():.2f}%")
        print(f"Division: {student.division()}\n")
```

```
Name: Alice
Roll Number: 101
Total Marks: 425
Percentage: 85.00%
Division: First Division

Name: Bob
Roll Number: 102
Total Marks: 375
Percentage: 75.00%
Division: First Division

Name: Charlie
Roll Number: 103
Total Marks: 325
Percentage: 65.00%
Division: First Division

Name: David
Roll Number: 104
Total Marks: 275
Percentage: 55.00%
Division: Second Division

Name: Eve
Roll Number: 105
Total Marks: 225
Percentage: 45.00%
Division: Second Division
```

32. . Create a parent class Bonus with instance variables sales_id and sales_amount.

- Add get_bonus method that calculates a salesperson's bonus using the foumula bonus = sales * 0.05.
- Create a child class named PremiumBonus from Bonus. The child class's get_premium_bonus() method should calculate the bonus using the formula bonus = sales * 0.05 + (sales – 2500) * 0.01.

Now, create an object of PremiumBonus class and use this object to find both bonus and premium bonus

```
class Bonus:
    def __init__(self, sales_id, sales_amount):
        self.sales_id = sales_id
        self.sales_amount = sales_amount

    def get_bonus(self):
        bonus = self.sales_amount * 0.05
        return bonus
```

```
class PremiumBonus(Bonus):
    def get_premium_bonus(self):
        premium_bonus = super().get_bonus() + (self.sales_amount - 2500) * 0.01
        return premium_bonus

# Create an object of PremiumBonus class
premium_bonus_obj = PremiumBonus("S123", 5000)

# Calculate and display both bonus and premium bonus
bonus = premium_bonus_obj.get_bonus()
premium_bonus = premium_bonus_obj.get_premium_bonus()

print(f"Bonus: {bonus}")
print(f"Premium Bonus: {premium_bonus}")
```

```
Bonus: 250.0
Premium Bonus: 275.0
```

```
### 34 . Write a program that reads a text file and displays the follo
* a. Number of characters
* b. Number of vowels
* c. Number of consonants
* d. Number of words
* e. Number of lines
```

34 . Write a program that reads a text file and displays the following:

- a. Number of characters
- b. Number of vowels
- c. Number of consonants
- d. Number of words
- e. Number of lines

```
def count_characters(text):
    return len(text)

def count_vowels(text):
    vowels = "AEIOUaeiou"
    count = 0
    for char in text:
        if char in vowels:
            count += 1
    return count

def count_consonants(text):
    consonants = "BCDFGHJKLMNPQRSTVWXYZbcdfghjklmnpqrstvwxyz"
    count = 0
    for char in text:
        if char in consonants:
            count += 1
    return count

def count_words(text):
    words = text.split()
    return len(words)

def count_lines(text):
    lines = text.split('\n')
    return len(lines)

# Read the uploaded text file
with open('/content/sample_data/ats.txt', 'r') as file:
    file_contents = file.read()

# Print the file contents (for debugging)
print("File Contents:")
print(file_contents)

# Calculate and display the requested information
num_characters = count_characters(file_contents)
num_vowels = count_vowels(file_contents)
num_consonants = count_consonants(file_contents)
num_words = count_words(file_contents)
num_lines = count_lines(file_contents)
```

```
print(f"Number of characters: {num_characters}")
print(f"Number of vowels: {num_vowels}")
print(f"Number of consonants: {num_consonants}")
print(f"Number of words: {num_words}")
print(f"Number of lines: {num_lines}")
```

```
File Contents:
Power monitoring function locally and remotely
Input and Output voltage monitoring
Load Current Monitoring
Local and Remote automatic alarm
Built in Web Interface


In the data center, it is vitally important to have uninterrupted power. To achieve this goal, an automatic transfer switch (ATS) is use
that automatically transfers a power supply from its primary source to a backup source when it senses a failure or outage in the primary
source. When a failure occurs in a primary power system, the ATS invokes a standby power source, such as an uninterruptable power
supply. An ATS can also start up more long-term backup power systems, such as local diesel generators, to run electric equipment until
utility power is restored.
Since the ATS is connected to both primary and backup power sources, it serves as an intermediary between equipment and the
power supplies, acting as an electrical relay. An ATS can also act as a redundant, rack-mounted power supply for equipment that is
connected to a power source with only one cord.
Canovate®
 ATS provides high reliable and redundant power to the attached equipment. Single fed servers and other telecom
equipment, small power supplies, our front & side coolers and many more devices can be connected to the ATS to gain more power
secure and redundant characteristics. The ATS has two input power sources, so if the primary source shut down, it will seamlessly
switch to the secondary power source automatically. Canovate®
 ATS has 2 basic models available, if the user is looking for remote
management, we can place the optional SNMP card to upgrade the basic type to the remote management, intelligent one.


Number of characters: 1669
Number of vowels: 527
Number of consonants: 825
Number of words: 272
```