Oops Assignment
Submitted by : Mayur Gadhave
DS060623

Challenge 1: Square Numbers and Return Their Sum

In [4]:

```
class Point:
    def __init__(self, x, y, z):
        self.x = x
        self.y = y
        self.z = z

    def sqSum(self):
        return self.x**2 + self.y**2 + self.z**2


x_value = int(input("Enter the value of x: "))
y_value = int(input("Enter the value of y: "))
z_value = int(input("Enter the value of z: "))

# Create a Point object with the user-provided values
point_object = Point(x_value, y_value, z_value)

# Call the sqSum() method to get the sum of squared properties
result = point_object.sqSum()

print("The sum of squares of given numbers is ", result)
```

```
Enter the value of x: 1
Enter the value of y: 3
Enter the value of z: 5
The sum of squares of given numbers is  35
```

Challenge 2: Implement a Calculator Class

In [8]:

```
class Calculator:
    def __init__(self, num1, num2):
        self.num1 = num1
        self.num2 = num2

    def add(self):
        return self.num1 + self.num2

    def subtract(self):
        return self.num2 - self.num1

    def multiply(self):
        return self.num1 * self.num2

    def divide(self):
        # Ensure num1 is not zero to avoid division by zero error
        if self.num1 == 0:
            raise ValueError("Cannot divide by zero.")
        return self.num2 / self.num1


num1_value = int(input("Enter the value of num1: "))
num2_value = int(input("Enter the value of num2: "))

# Creating a Calculator object with the user-provided values
obj = Calculator(num1_value, num2_value)

# Performing the calculations using the methods of the Calculator class
addition_result = obj.add()
subtraction_result = obj.subtract()
multiplication_result = obj.multiply()

print("Addition Result:", addition_result)
print("Subtraction Result:", subtraction_result)
print("Multiplication Result:",multiplication_result)

try:
    division_result = obj.divide()
    print("Division Result:",division_result)
except ValueError as e:
    print(e)
```

```
Enter the value of num1: 10
Enter the value of num2: 94
Addition Result: 104
Subtraction Result: 84
Multiplication Result: 940
Division Result: 9.4
```

Challenge 3: Implement the Complete Student Class

In [11]:

```
class Student:
    def setName(self, name):
        self.__name = name

    def setRollNumber(self, rollNumber):
        self.__rollNumber = rollNumber

    def getName(self):
        return self.__name

    def getRollNumber(self):
        return self.__rollNumber


# Creating a Student object and setting the properties using the setter methods
student = Student()
student.setName("Mayur Gadhave")
student.setRollNumber(322)

# Accessing the private properties using the getter methods
print("Name:", student.getName())
print("Roll Number:", student.getRollNumber())
```

```
Name: Mayur Gadhave
Roll Number: 322
```

Challenge 4: Implement a Banking Account

In [13]:

```
class Account:
    def __init__(self, title=None, balance=0):
        self.title = title
        self.balance = balance


class SavingsAccount(Account):
    def __init__(self, title=None, balance=0, interestRate=0):
        super().__init__(title, balance)
        self.interestRate = interestRate


# Creating an instance of the SavingsAccount class with the given parameters
savings_account = SavingsAccount("Ashish", 5000, 5)

# Printing the properties
print("Title:", savings_account.title)
print("Balance:", savings_account.balance)
print("Interest Rate:", savings_account.interestRate)
```

```
Title: Ashish
Balance: 5000
Interest Rate: 5
```

Challenge 5: Handling a Bank Account

In [16]:

```
class Account:
    def __init__(self, title=None, balance=0):
        self.title = title
        self.balance = balance

    def getBalance(self):
        return self.balance

    def deposit(self, amount):
        self.balance += amount

    def withdrawal(self, amount):
        self.balance -= amount


class SavingsAccount(Account):
    def __init__(self, title=None, balance=0, interestRate=0):
        super().__init__(title, balance)
        self.interestRate = interestRate

    def interestAmount(self):
        return (self.interestRate * self.balance) / 100


savings_account = SavingsAccount("Aashish", 10000, 10)

print("Initial Balance:", savings_account.getBalance())

interest_amount = savings_account.interestAmount()
print("Interest Amount:", interest_amount)

# Depositing some amount to the savings account
savings_account.deposit(2000)
print("After Deposit:", savings_account.getBalance())

interest_amount = savings_account.interestAmount()
print("Interest Amount after Deposit:", interest_amount)
```

```
Initial Balance: 10000
Interest Amount: 1000.0
After Deposit: 12000
Interest Amount after Deposit: 1200.0
```