Assignment No. 4 & 5
Class, Object, Constructor & Constructor chaining

1.Create a class named 'Student' with String variable 'name' and integer variable 'roll_no'. Assign the value of roll_no as '2' and that of name as "John" by creating an object of the class Student.

2. Assign and print the roll number, phone number and address of two students having names "Sam" and "John" respectively by creating two objects of class 'Student'.

3. Write a program to print the area and perimeter of a triangle having sides of 3, 4 and 5 units by creating a class named 'Triangle' without any parameter in its constructor.

4. Write a program to print the area and perimeter of a triangle having sides of 3, 4 and 5 units by creating a class named 'Triangle' with the constructor having the three sides as its parameters.

5. Write a program to print the area of two rectangles having sides (4,5) and (5,8) respectively by creating a class named 'Rectangle' with a method named 'Area' which returns the area and length and breadth passed as parameters to its constructor.

6. Write a program to print the area of a rectangle by creating a class named 'Area' having two methods. First method named as 'setDim' takes length and breadth of the rectangle as parameters and the second method named as 'getArea' returns the area of the rectangle. Length and breadth of the rectangle are entered through the keyboard.

7. Write a program to print the area of a rectangle by creating a class named 'Area' taking the values of its length and breadth as parameters of its constructor and having a method named 'returnArea' which returns the area of the rectangle. Length and breadth of rectangle are entered through the keyboard.

8. Print the average of three numbers entered by the user by creating a class named 'Average' having a method to calculate and print the average.

9. Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by the user.

10. Write a program that would print the information (name, year of joining, salary, address) of three employees by creating a class named 'Employee'. The output should be as follows:

```
Name      Year of joining      Address

Robert         1994           64C- WallsStreat

Sam            2000           68D- WallsStreat

John           1999           26B- WallsStreat
```

Write a Java class Clock for dealing with the day time represented by hours, minutes, and seconds. Your class must have the following features:

- Three instance variables for the hours (range 0 - 23), minutes (range 0 - 59), and seconds (range 0 - 59).
- Three constructors:

  default (with no parameters passed; is should initialize the represented time to 12:0:0)

  a constructor with three parameters: hours, minutes, and seconds.

  a constructor with one parameter: the value of time in seconds since midnight (it should be converted into the time value in hours, minutes, and seconds)
- Instance methods:

  a *set*-method method setClock() with one parameter *seconds* since midnight (to be converted into the time value in hours, minutes, and seconds as above).

  get-methods getHours(), getMinutes(), getSeconds() with no parameters that return the corresponding values.

  set-methods setHours(), setMinutes(), setSeconds() with one parameter each that set up the corresponding instance variables.

method tick() with no parameters that increments the time stored in a Clock object by one second.
method addClock() accepting an object of type Clock as a parameter. The method should add the time represented by the parameter class to the time represented in the current class.
Add an instance method toString() with no parameters to your class. toString() must return a String representation of the Clock object in the form "(hh:mm:ss)", for example "(03:02:34)".
Add an instance method tickDown() which decrements the time stored in a Clock object by one second.
Add an instance method subtractClock() that takes one Clock parameter and returns the difference between the time represented in the current Clock object and the one represented by the Clock parameter. Difference of time should be returned as a clock object.

Write a separate class ClockDemo with a main() method. The program should:

- instantiate a Clock object firstClock using one integer *seconds* since midnight obtained from the keyboard.
- tick the clock ten times by applying its *tick()* method and print out the time after each tick.
- Extend your code by appending to it instructions instantiating a Clock object secondClock by using three integers (hours, minutes, seconds) read from the keyboard.
- Then tick the clock ten times, printing the time after each tick.
- Add the secondClock time in firstClock by calling the method addClock.
- Print both clock objects calling toString method

Create a reference thirdClock that should reference the object of difference of firstClock and secondClock by calling the method subtractClock().

Question 3

Write a Java class Complex for dealing with complex numbers. Your class must have the following features:

- Instance variables :

realPart for the real part of type double

imaginaryPart for imaginary part of type double.

- Constructor:

  public Complex (): A default constructor, it should initialize the number to 0, 0)

  public Complex (double realPart, double imaginaryPart): A constructor with parameters, it creates the complex object by setting the two fields to the passed values.

- Instance methods:

  public Complex add (Complex otherNumber): This method will find the sum of the current complex number and the passed complex number. The method returns a new Complex number which is the sum of the two.

  public Complex subtract (Complex otherNumber): This method will find the difference of the current complex number and the passed complex number. The method returns a new Complex number which is the difference of the two.

  public Complex multiply (Complex otherNumber): This method will find the product of the current complex number and the passed complex number. The method returns a new Complex number which is the product of the two.

  public Complex divide (Complex otherNumber): This method will find the ... of the current complex number and the passed complex number. The method returns a new Complex number which is the ... of the two.

  public void setRealPart (double realPart): Used to set the real part of this complex number.

  public void setImaginaryPart (double realPart): Used to set the imaginary part of this complex number.

  public double getRealPart(): This method returns the real part of the complex number

  public double getImaginaryPart(): This method returns the imaginary part of the complex number

  public String toString(): This method allows the complex number to be easily printed out to the screen

Write a separate class **ComplexDemo** with a main() method and test the Complex class methods.

Question 4

Write a Java class Author with following features:

- Instance variables :

  firstName for the author's first name of type String.

  lastName for the author's last name of type String.
- Constructor:

  public Author (String firstName, String lastName): A constructor with parameters, it creates the Author object by setting the two fields to the passed values.
- Instance methods:

  public void setFirstName (String firstName): Used to set the first name of the author.

  public void setLastName (String lastName): Used to set the last name of the author.

  public double getFirstName(): This method returns the first name of the author.

  public double getLastName(): This method returns the last name of the author.

  public String toString(): This method printed out author's name to the screen

Write a Java class Book with following features:

- Instance variables :

  title for the title of book of type String.

  author for the author's name of type String.

  price for the book price of type double.
- Constructor:

  public Book (String title, Author name, double price): A constructor with parameters, it creates the Author object by setting the fields to the passed values.
- Instance methods:

public void setTitle(String title): Used to set the title of a book.
public void setAuthor(String author): Used to set the name of the author of a book.
public void setPrice(double price): Used to set the price of a book.
public double getTitle(): This method returns the title of the book.
public double getAuthor(): This method returns the author's name of the book.

public String toString(): This method printed out book's details to the screen

Write a separate class **BookDemo** with a main() method creates a Book titled "Developing Java Software" with authors Russel Winderand price 79.75. Prints the Book's string representation to standard output (using System.out.println).

1.Write a program by creating an 'Employee' class having the following methods and print the final salary.

1 - 'getInfo()' which takes the salary, number of hours of work per day of employee as parameter

2 - 'AddSal()' which adds $10 to the salary of the employee if it is less than $500.

3 - 'AddWork()' which adds $5 to the salary of an employee if the number of hours of work per day is more than 6 hours.

2. Create a class called 'Matrix' containing a constructor that initializes the number of rows and number of columns of a new Matrix object. The Matrix class has the following information:

1 - number of rows of matrix

2 - number of columns of matrix

3 - elements of matrix in the form of 2D array

3.The Matrix class has methods for each of the following:

1 - get the number of rows

2 - get the number of columns

3 - set the elements of the matrix at given position (i,j)

4 - adding two matrices. If the matrices are not addable, "Matrices cannot be added" will be displayed.

5 - multiplying the two matrices

****

1. Constructor Chaining within the Same Class

Create a class Car with multiple constructors that initialize different attributes using constructor chaining.

Problem Statement:

- Create a class Car with attributes brand, model, and price.
- Implement **constructor chaining** within the same class:
    - One constructor should only take the brand.
    - Another constructor should take brand and model.
    - The final constructor should take brand, model, and price.
- Use the this() keyword to call other constructors.
- Display car details in each constructor.

✅ **Task:** Create objects using different constructors and observe constructor chaining in action.

2. Constructor Chaining Using super Keyword (Parent-Child Relationship)

Create a class hierarchy where the child class calls the parent class constructor using super().

Problem Statement:

- Create a Person class with attributes name and age.
- Create a Student class that extends Person and has an additional attribute course.
- Use constructor chaining:
    - Person class should have a constructor initializing name and age.
    - Student class should use super(name, age) to call the Person constructor and then initialize course.
- Display details in both constructors.

✅ **Task:** Create a Student object and verify that both constructors (parent and child) are executed in sequence.

3. Multi-Level Constructor Chaining (Grandparent → Parent → Child)

Demonstrate constructor chaining in a multi-level inheritance scenario.

Problem Statement:

- Create a Vehicle class with an attribute type.
- Create a subclass FourWheeler with an additional attribute brand.
- Create another subclass Car with attributes model and price.
- Use **multi-level constructor chaining**:
  - Vehicle initializes type.
  - FourWheeler calls super(type) and initializes brand.
  - Car calls super(type, brand), initializes model, and price.
- Display details at each level.

✅ **Task:** Create a Car object and verify that constructors are executed from parent → child → grandchild.

4. Constructor Chaining in an E-Commerce Scenario

Create an Order class where different constructors initialize order details using constructor chaining.

Problem Statement:

- Create an Order class with attributes orderId, customerName, and totalAmount.
- Implement **constructor chaining** within the same class:
  - One constructor initializes only orderId.
  - Another constructor initializes orderId and customerName by calling the first constructor.
  - The final constructor initializes all three attributes (orderId, customerName, totalAmount) by calling the second constructor.
- Display order details.

✅ **Task:** Create Order objects using different constructors and verify how chaining works.

5. Constructor Chaining in a Banking System

Create a BankAccount class where constructor chaining initializes different account types.

Problem Statement:

- Create a BankAccount class with attributes accountNumber, holderName, and balance.
- Implement **constructor chaining**:
  - One constructor initializes only accountNumber.
  - Another constructor initializes accountNumber and holderName, calling the first constructor.
  - The final constructor initializes all three attributes by calling the second constructor.
- Implement a method to **display account details**.

✅ **Task:** Create BankAccount objects using different constructors and verify the constructor call sequence.