1. **(this and getter)**
   Define a Person class. It should have two attributes: name and age. Write getter methods using “this” to return these variables. The main is given and output should look like below.

```java
public class Main {
    public static void main(String[] args) {
        Person person = new Person("Alice", 30);
        System.out.println("Name: " + person.getName());
        System.out.println("Age: " + person.getAge());
    }
}
```

```
Name: Alice
Age: 30
```

2. **(Attributes and constructors)**
   Write a Java class to create a class called "Dog" with a name and breed attribute. Create two instances of the "Dog" class, set their attributes using the constructor and modify the attributes using the setter methods and print the updated values. The main is given and output should look like below.

```java
public class Main {
    public static void main(String[] args) {
        Dog dog1 = new Dog("Buddy", "Golden Retriever");
        Dog dog2 = new Dog("Max", "Beagle");

        System.out.println("Initial Details:");
        dog1.printDetails();
        dog2.printDetails();

        dog1.setName("Charlie");
        dog1.setBreed("Labrador");
        dog2.setName("Rocky");
        dog2.setBreed("Bulldog");

        // Print updated details
        System.out.println("\nUpdated Details:");
        dog1.printDetails();
        dog2.printDetails();
    }
}
```

```
Initial Details:
Name: Buddy
Breed: Golden Retriever
Name: Max
Breed: Beagle

Updated Details:
Name: Charlie
Breed: Labrador
Name: Rocky
Breed: Bulldog
```

3. **(Methods, attributes and classes)**
   Write a Java program to create a class called "TrafficLight" with attributes for color and duration, and methods to change the color and check for red or green. The main is given and output should look like below.

```java
public class Main {
    public static void main(String[] args) {
        TrafficLight light = new TrafficLight("Red", 60);

        System.out.println("Initial Traffic Light Details:");
        light.printDetails();

        light.changeColor("Green");
        light.setDuration(30);

        System.out.println("\nUpdated Traffic Light Details:");
        light.printDetails();

        System.out.println("\nIs the light red? " + light.isRed());
        System.out.println("Is the light green? " + light.isGreen());
    }
}
```

```
Initial Traffic Light Details:
Color: Red
Duration: 60 seconds

Updated Traffic Light Details:
Color: Green
Duration: 30 seconds

Is the light red? false
Is the light green? true
```

Question 4, 5, 6, 7, 8 and 9 are connected

4. **(Methods, attributes and classes)**
   Define the class and constructor for the Restaurant class. The constructor should initialize the arrays for menu items, prices, and ratings with the specified capacity and set the size to 0. The main is given and output should look like below.

```
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        System.out.println("Restaurant created with capacity 10.");
    }
}
```

```
Restaurant created with capacity 10.
```

5. **(Methods, attributes and classes)**
   Write the addMenuItem method for the Restaurant class. This method should add a new menu item with its price to the arrays if there is space available. If the menu is full, it should print a message indicating that no more items can be added. The main is given and output should look like below.

```
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        restaurant.addMenuItem("Burger", 5.99);
        restaurant.addMenuItem("Pizza", 8.99);
    }
}
```

```
Menu:
Burger: $5.99, Rating: 0.0
Pizza: $8.99, Rating: 0.0
```

6. **(Methods, attributes and classes)**
   Write the printMenu method which will give the following output. In this case, as you haven't done the addRating method, you should expect the Rating as 0. The main is given and output should look like below.

```
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        restaurant.addMenuItem("Burger", 5.99);
        restaurant.addMenuItem("Pizza", 8.99);
        restaurant.printMenu();
    }
}
```

```
Burger: $5.99, Rating: 0
Pizza: $8.99, Rating: 0
Salad: $4.99, Rating: 0
```

7. **(Methods, attributes and classes)**
   Write the removeMenuItem method for the Restaurant class. This method should find the item to be removed, shift all subsequent items one position to the left, and reduce the size of the menu by 1. It should also handle cases where the item is not found. Use an array. You may need additional methods (such as findIndexItem{} which take itemName as parameter). The main is given below.

```
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        restaurant.removeMenuItem("Pizza");
        }
}
```

8. **(Methods, attributes and classes)**
   Write the addRating method for the Restaurant class. This method should update the rating of an existing menu item if it is found. If the item is not on the menu, it should print a message indicating that the item was not found. The main is given below.

```java
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        restaurant.addMenuItem("Burger", 5.99);
        restaurant.addRating("Burger", 4.5);
    }
}
```

9. **(Methods, attributes and classes)**
   Write the calculateAverageRating method for the Restaurant class. This method should calculate the average rating of all menu items with positive ratings. If there are no items or no positive ratings, it should return 0.0. The main is given and output should look like below.

```java
public class Main {
    public static void main(String[] args) {
        Restaurant restaurant = new Restaurant(10);
        restaurant.addMenuItem("Burger", 5.99);
        restaurant.addMenuItem("Pizza", 8.99);
        restaurant.addRating("Burger", 4.5);
        restaurant.addRating("Pizza", 4.8);

        double averageRating = restaurant.calculateAverageRating();
        System.out.printf("Average Rating: %.1f\n", averageRating);
    }
}
```

```
Average Rating: 4.7
```

10. **(this and constructors)**
    Create a Circle class with two constructors. One constructor should take radius, while the other should take no parameters and call the first constructor using this. Describe the purpose of using this in this case. The main is given and output should look like below.

```java
public class Main {
    public static void main(String[] args) {
        Circle defaultCircle = new Circle();  // Uses the default constructor
        Circle customCircle = new Circle(5.0);  // Uses the constructor with radius

        System.out.println("Default Circle Radius: " + defaultCircle.getRadius()); // Output: 1.0
        System.out.println("Custom Circle Radius: " + customCircle.getRadius());  // Output: 5.0
    }
}
```

```
Default Circle Radius: 1.0
Custom Circle Radius: 5.0
```

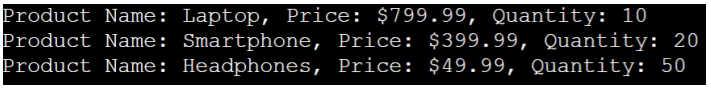
11. **(Array)**
    Write a Java class with the method printSquare. There should be only the printSquare method with no extra constructor and attributes. The main is given and output should look like below.

```
*****
*****
*****
*****
*****
```

```java
public class Main {
    public static void main(String[] args) {
        SquareOfStars square = new SquareOfStars();
        int size = 5;
        square.printSquare(size);
    }
}
```

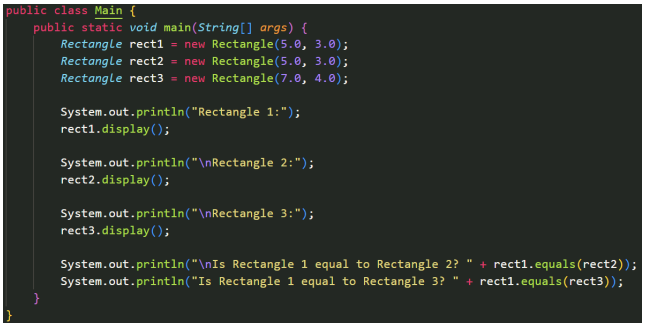
12. **(Array, constructors and attributes)**
    Write a Java class to create a class called Product with attributes for productName, price, and quantity. Use an array to store multiple Product objects. Implement methods to add a product and display all products. The main is given and output should look like below.

```
public class Main {
    public static void main(String[] args) {
        Product[] catalog = new Product[3];

        catalog[0] = new Product("Laptop", 799.99, 10);
        catalog[1] = new Product("Smartphone", 399.99, 20);
        catalog[2] = new Product("Headphones", 49.99, 50);

        for (Product product : catalog) {
            product.display();
        }
    }
}
```

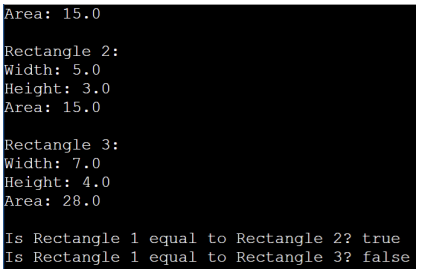


```
Product Name: Laptop, Price: $799.99, Quantity: 10
Product Name: Smartphone, Price: $399.99, Quantity: 20
Product Name: Headphones, Price: $49.99, Quantity: 50
```

13. **(Constructors and attributes)**
    Implement the Rectangle class in Rectangle.java.You will write the constructor, also the class will have the constructor. Your rectangle class should have getArea, getWidth and getHeight getters and you will need an equals method. In Main.java, create three Rectangle objects with the following dimensions:
    Rectangle 1: width = 5.0, height = 3.0
    Rectangle 2: width = 5.0, height = 3.0
    Rectangle 3: width = 7.0, height = 4.0
    Use the display method to print the details of each rectangle and compare their areas using the equals method. You will use the “this” keyword. In the rectangle class you will have width and height attributes. The main is given and output should look like below. Your attributes will be width and height.



```
public class Main {
    public static void main(String[] args) {
        Rectangle rect1 = new Rectangle(5.0, 3.0);
        Rectangle rect2 = new Rectangle(5.0, 3.0);
        Rectangle rect3 = new Rectangle(7.0, 4.0);

        System.out.println("Rectangle 1:");
        rect1.display();

        System.out.println("\nRectangle 2:");
        rect2.display();

        System.out.println("\nRectangle 3:");
        rect3.display();

        System.out.println("\nIs Rectangle 1 equal to Rectangle 2? " + rect1.equals(rect2));
        System.out.println("Is Rectangle 1 equal to Rectangle 3? " + rect1.equals(rect3));
    }
}
```



```
Area: 15.0

Rectangle 2:
Width: 5.0
Height: 3.0
Area: 15.0

Rectangle 3:
Width: 7.0
Height: 4.0
Area: 28.0

Is Rectangle 1 equal to Rectangle 2? true
Is Rectangle 1 equal to Rectangle 3? false
```

Question 14, 15 and 16 are connected

14. **(Constructors and Method)**
    You will create a Sudoku constructor, you will create the initializeBoard() method. This method should initialize all cells of the 4x4 Sudoku board to zero. You will need a constant value to hold board “int SIZE” and the matrix “int [][] board” for this question. Also you will use arrays.

```
public class Main {
    public static void main(String[] args) {
        Sudoku sudoku = new Sudoku();
        sudoku.initializeBoard();
    }
}
```

15. **(Constructors and Method)**
    In the Sudoku class, implement the fillBoard() method. This method should fill the 4x4 Sudoku board with random numbers between 1 and 4. Also you will use arrays. Also you will use arrays.

```
public class Main {
    public static void main(String[] args) {
        Sudoku sudoku = new Sudoku();
        sudoku.initializeBoard();
        sudoku.fillBoard();
    }
}
```

16. **(Constructors and Method)**
    In the Sudoku class, write the printBoard() method. This method should print all the numbers in the Sudoku board to the screen in a properly formatted manner. Also you will use arrays.

```
public class Main {
    public static void main(String[] args) {
        Sudoku sudoku = new Sudoku();
        sudoku.initializeBoard();
        sudoku.fillBoard();
        sudoku.printBoard();
    }
}
```

17. **(Array and Method)**
    Create a Java class Fibonacci with a method generateFibonacci(int n) that generates the first 10 Fibonacci numbers and returns them as an array of integers. You do not need attributes, constructor. Just write the class which will only have the method. Also you will use arrays.

```java
public class Main {
    public static void main(String[] args) {
        Fibonacci fibonacci = new Fibonacci();

        int[] firstTenFibonacci = fibonacci.generateFibonacci(10);
        System.out.println("First 10 fibonacci numbers:");
        for (int num : firstTenFibonacci) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```