BITA/7/23/-49/TZ

QUESTION 1

(Rectangle Class) Create a class Rectangle with attributes length and width, each of which defaults to 1. Provide member functions that calculate the perimeter and the area of the rectangle. Also, provide set and get functions for the length and width attributes. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0

PROGRAM AND OUTPUT

```
class Rectangle {
    private double length = 1.0;
    private double width = 1.0;

    public void setLength(double length) {
        if (length > 0.0 && length < 20.0) {
            this.length = length;
        } else {
            System.out.println("Invalid length value!");
        }
    }

    public void setWidth(double width) {
        if (width > 0.0 && width < 20.0) {
            this.width = width;
        } else {
            System.out.println("Invalid width value!");
        }
    }

    public double getLength() {
        return length;
    }
```

PROGRAM AND OUTPUT

```
25      public double getWidth() {
26          return width;
27      }
28
29      public double calculatePerimeter() {
30          return 2 * (length + width);
31      }
32
33      public double calculateArea() {
34          return length * width;
35      }
36
37      public static void main(String[] args) {
38          Rectangle rectangle = new Rectangle();
39          rectangle.setLength(15.5);
40          rectangle.setWidth(10.2);
41          System.out.println("Length: " + rectangle.getLength());
42          System.out.println("Width: " + rectangle.getWidth());
43          System.out.println("Perimeter: " + rectangle.calculatePerimeter());
44          System.out.println("Area: " + rectangle.calculateArea());
45      }
46  }
47
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\CHARMING\desktop\kazz> java Rectangle
Length: 15.5
Width: 10.2
Perimeter: 51.4
Area: 158.1
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 2

. (Invoice Class) Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as data members a part number (type string), a part description (type string), a quantity of the item being purchased (type int) and a price per item (type int). Your class should have a constructor that initializes the four data members. Provide a set and a get function for each data member. In addition, provide a member function named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as an int value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0. Write a test program that demonstrates class Invoice's capabilities

PROGRAM

```
class Invoice {
    private String partNumber;
    private String partDescription;
    private int quantity;
    private int pricePerItem;

    public Invoice(String partNumber, String partDescription, int quantity, int pricePerItem) {
        this.partNumber = partNumber;
        this.partDescription = partDescription;
        this.quantity = Math.max(0, quantity);
        this.pricePerItem = Math.max(0, pricePerItem);
    }

    public void setPartNumber(String partNumber) {
        this.partNumber = partNumber;
    }

    public void setPartDescription(String partDescription) {
        this.partDescription = partDescription;
    }

    public void setQuantity(int quantity) {
        this.quantity = Math.max(0, quantity);
    }

    public void setPricePerItem(int pricePerItem) {
        this.pricePerItem = Math.max(0, pricePerItem);
    }

    public String getPartNumber() {
        return partNumber;
    }
```

```java
    public String getPartDescription() {
        return partDescription;
    }

    public int getQuantity() {
        return quantity;
    }

    public int getPricePerItem() {
        return pricePerItem;
    }

    public int getInvoiceAmount() {
        return quantity * pricePerItem;
    }

    public static void main(String[] args) {
        Invoice invoice = new Invoice("001", "Hammer", 5, 20);
        System.out.println("Part Number: " + invoice.getPartNumber());
        System.out.println("Description: " + invoice.getPartDescription());
        System.out.println("Quantity: " + invoice.getQuantity());
        System.out.println("Price Per Item: $" + invoice.getPricePerItem());
        System.out.println("Total Amount: $" + invoice.getInvoiceAmount());
    }
}
```

OUTPUT

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Invoice.java
PS C:\Users\CHARMING\desktop\kazz> java  Invoice
Part Number: 001
Description: Hammer
Quantity: 5
Price Per Item: $20
Total Amount: $100
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 3

(Account Class) Create a class called Account that a bank might use to represent customers' bank accounts. Your class should include one data member of type int to represent the account balance. Your class should provide a constructor that receives an initial balance and uses it to initialize the data member. The constructor should validate the initial balance to ensure that it is greater than or equal to 0. If not, the balance should be set to 0 and the constructor should display an error message, indicating that the initial balance was invalid. The class should provide three member functions. Member function credit should add an amount to the current balance. Member function debit should withdraw money from the Account and should ensure that the debit amount does not exceed the Account's balance. If it does, the balance should be left unchanged and the function should print a message indicating "Debit amount exceeded account balance." Member function getBalance should return the current balance. Create a program that creates two Account objects and tests the member functions of class Account.

PROGRAM

```java
class Employee {
    private String firstName;
    private String lastName;
    private int monthlySalary;

    public Employee(String firstName, String lastName, int monthlySalary) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.monthlySalary = Math.max(monthlySalary, 0);
    }
    public int getYearlySalary() {
        return monthlySalary * 12;
    }

    public void giveRaise(double percentage) {
        monthlySalary += (int)(monthlySalary * percentage / 100);
    }

    public static void main(String[] args) {
        Employee emp1 = new Employee("John", "Doe", 3000);
        Employee emp2 = new Employee("Jane", "Smith", 4000);

        System.out.println("Yearly Salary of Employee 1: $" + emp1.getYearlySalary());
        System.out.println("Yearly Salary of Employee 2: $" + emp2.getYearlySalary());

        emp1.giveRaise(10);
        emp2.giveRaise(10);

        System.out.println("Yearly Salary of Employee 1 after raise: $" + emp1.getYearlySalary());
        System.out.println("Yearly Salary of Employee 2 after raise: $" + emp2.getYearlySalary());
    }
}
```

OUTPUT

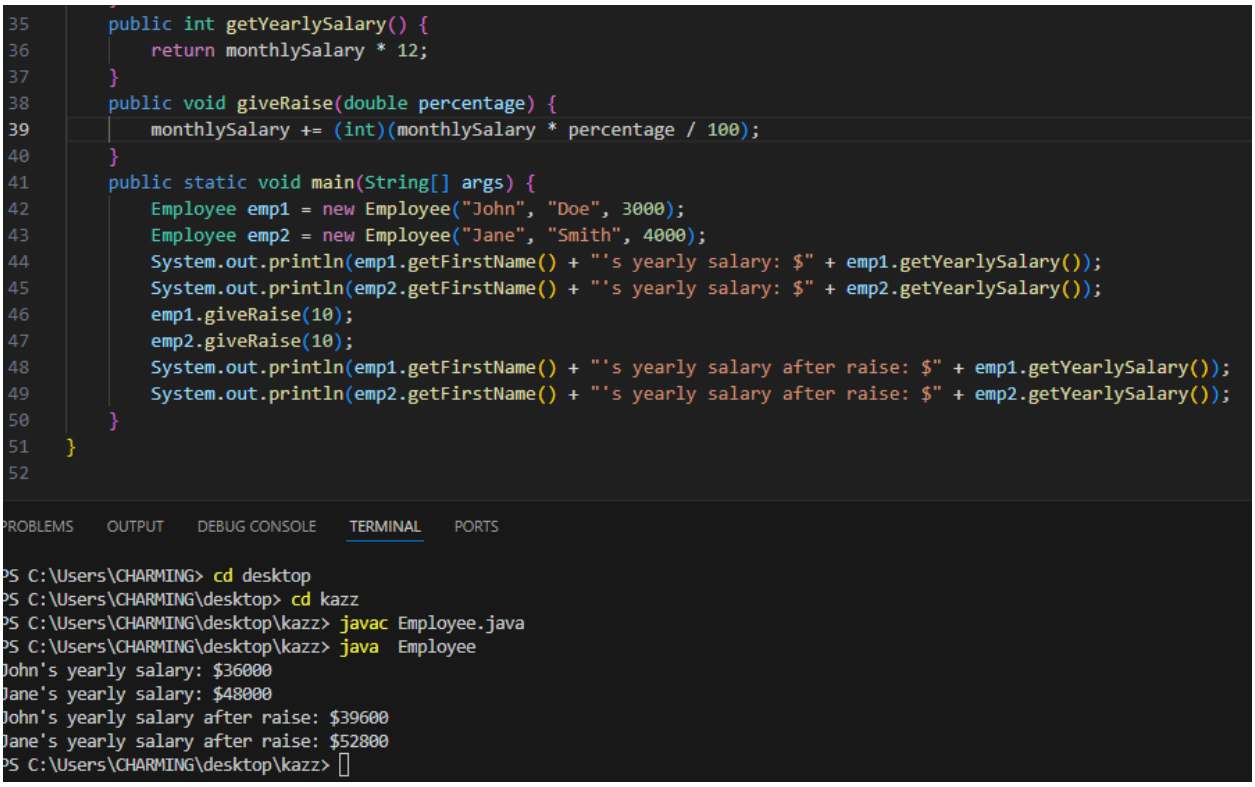
```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Account.java
PS C:\Users\CHARMING\desktop\kazz> java Account
Error: Initial balance invalid.
Account 1 Balance: $120
Debit amount exceeded account balance.
Account 2 Balance: $100
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 4

(Employee Class) Create a class called Employee that includes three pieces of information as data members a first name (type string), a last name (type string) and a monthly salary (type int). Your class should have a constructor that initializes the three data members. Provide a set and a get function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10 percent raise and display each Employee's yearly salary again.

PROGRAM AND OUTPUT

```java
class Employee {
    private String firstName;
    private String lastName;
    private int monthlySalary;
    public Employee(String firstName, String lastName, int monthlySalary) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.monthlySalary = Math.max(0, monthlySalary);
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public void setMonthlySalary(int monthlySalary) {
        this.monthlySalary = Math.max(0, monthlySalary);
    }
    public String getFirstName() {
        return firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public int getMonthlySalary() {
        return monthlySalary;
    }
    public int getYearlySalary() {
        return monthlySalary * 12;
    }
    public void giveRaise(double percentage) {
        monthlySalary += (int)(monthlySalary * percentage / 100);
    }
```



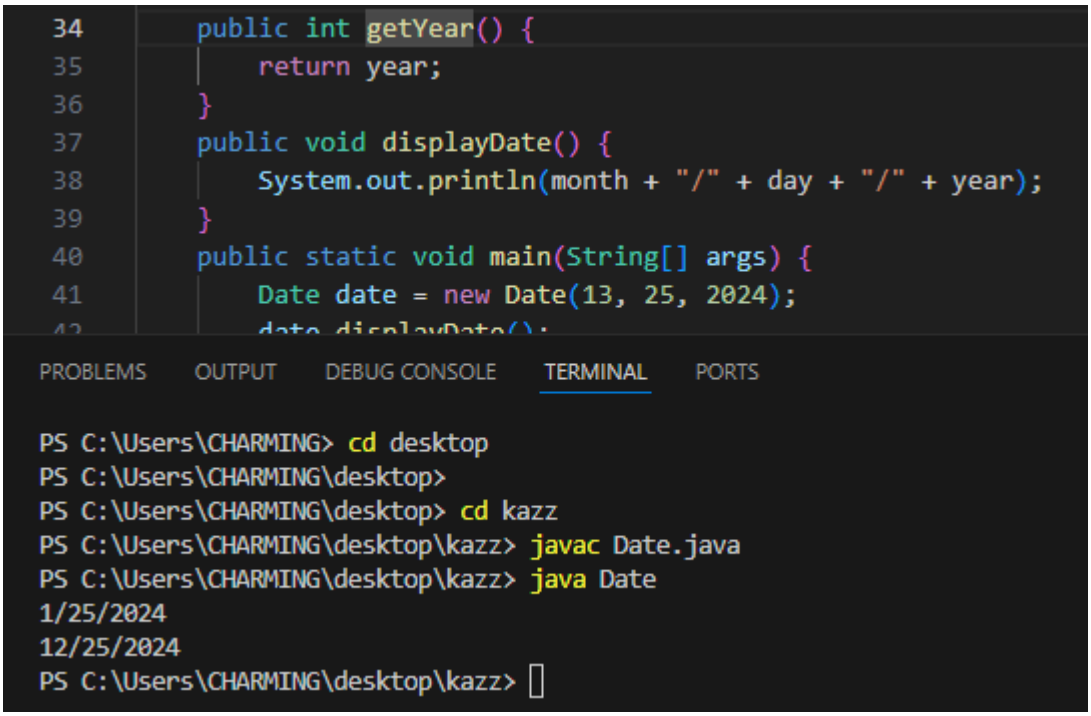
```
    public int getYearlySalary() {
        return monthlySalary * 12;
    }
    public void giveRaise(double percentage) {
        monthlySalary += (int)(monthlySalary * percentage / 100);
    }
    public static void main(String[] args) {
        Employee emp1 = new Employee("John", "Doe", 3000);
        Employee emp2 = new Employee("Jane", "Smith", 4000);
        System.out.println(emp1.getFirstName() + "'s yearly salary: $" + emp1.getYearlySalary());
        System.out.println(emp2.getFirstName() + "'s yearly salary: $" + emp2.getYearlySalary());
        emp1.giveRaise(10);
        emp2.giveRaise(10);
        System.out.println(emp1.getFirstName() + "'s yearly salary after raise: $" + emp1.getYearlySalary());
        System.out.println(emp2.getFirstName() + "'s yearly salary after raise: $" + emp2.getYearlySalary());
    }
}

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Employee.java
PS C:\Users\CHARMING\desktop\kazz> java  Employee
John's yearly salary: $36000
Jane's yearly salary: $48000
John's yearly salary after raise: $39600
Jane's yearly salary after raise: $52800
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 5

(Date Class) Create a class called Date that includes three pieces of information as data members a month (type int), a day (type int) and a year (type int). Your class should have a constructor with three parameters that uses the parameters to initialize the three data members. For the purpose of this exercise, assume that the values provided for the year and day are correct, but ensure that the month value is in the range 1-12; if it is not, set the month to 1. Provide a set and a get function for each data member. Provide a member function displayDate that displays the month, day and year separated by forward slashes (/). Write a test program that demonstrates class Date's capabilities.

PROGRAM AND OUTPUT

```java
class Date {
    private int month;
    private int day;
    private int year;

    public Date(int month, int day, int year) {
        this.month = (month >= 1 && month <= 12) ? month : 1;
        this.day = day;
        this.year = year;
    }

    public void setMonth(int month) {
        if (month >= 1 && month <= 12) {
            this.month = month;
        } else {
            System.out.println("Invalid month! Setting to 1.");
            this.month = 1;
        }
    }

    public void setDay(int day) {
        this.day = day;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public int getMonth() {
        return month;
    }

```



QUESTION 6

Create a class called Kitten that has three fields: String name, String owner, and int age. Create a constructor for Kitten that takes a String name and owner for the Kitten and uses them for initialization. Have the age for a Kitten start at 0; Implement a method for age which return a Kitten's age. Implement a method called haveBirthday that does not return anything and simply increases a Kitten's age by one. Finally, write a method called toString that returns a string of the form: " is and belongs to " e.g. "Bob is 87 and belongs to Gregor Samsa". Create an object which initialize three data members and print out the string above

PROGRAM

C: > Users > CHARMING > Desktop > kazz > Kitten.java

```java
class Kitten {
    private String name;
    private String owner;
    private int age;

    public Kitten(String name, String owner) {
        this.name = name;
        this.owner = owner;
        this.age = 0;
    }

    public int getAge() {
        return age;
    }

    public void haveBirthday() {
        age++;
    }

    @Override
    public String toString() {
        return name + " is " + age + " years old and belongs to " + owner;
    }

    public static void main(String[] args) {
        Kitten kitten = new Kitten("Whiskers", "John");
        System.out.println(kitten);
        kitten.haveBirthday();
        System.out.println(kitten);
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Kitten.java
PS C:\Users\CHARMING\desktop\kazz> java Kitten
Whiskers is 0 years old and belongs to John
Whiskers is 1 years old and belongs to John
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 7

Define a class called Triangle with three integer data members a, b, and c as the lengths of its three edges. This class should have the following methods: (a)a constructor with 3 parameters representing the 3 edges (b)a method isTriangle() which returns true if the 3 edges are all positive and they satisfy the triangle inequality where a+b > c, a+c > b, b+c > a. (c)a method getArea() which returns the area of triangle. NB: Area=1/2(a*b) The signature of these methods are given below: public Triangle(int newa, int newb, int newc) public boolean isTriangle() public double getArea() Note: getAngle() should return zero if the triangle is not really a triangle

PROGRAM AND OUTPUT

C: > Users > CHARMING > Desktop > kazz > Triangle.java

```java
class Triangle {
    private int a, b, c;
    public Triangle(int a, int b, int c) {
        this.a = a;
        this.b = b;
        this.c = c;
    }
    public boolean isTriangle() {
        return a > 0 && b > 0 && c > 0 && (a + b > c) && (a + c > b) && (b + c > a);
    }
    public double getArea() {
        if (!isTriangle()) return 0;
        double s = (a + b + c) / 2.0;
        return Math.sqrt(s * (s - a) * (s - b) * (s - c));
    }
    public static void main(String[] args) {
        Triangle triangle = new Triangle(3, 4, 5);
        System.out.println("Is triangle: " + triangle.isTriangle());
        System.out.println("Area: " + triangle.getArea());
    }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Triangle.java
PS C:\Users\CHARMING\desktop\kazz> java Triangle
Is triangle: true
Area: 6.0
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 8

Create class SavingsAccount. Usea static variable annualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a private instance variable savingsBalance indicating the amount the saver currently has ondeposit. Provide method calculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance by annualInterestRate divided by 12 this interest

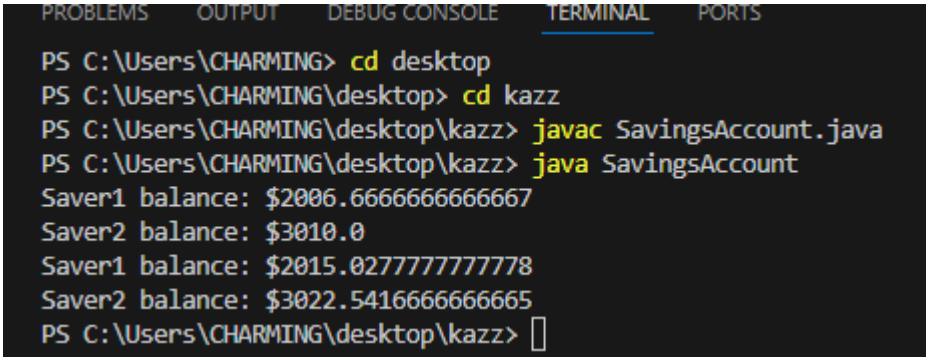
should be added to savingsBalance. Provide a static method modifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest and print the new balances for both savers. Then set the annualInterestRate to 5%, calculate the next month's interest and print the new balances for both savers.

PROGRAM

C: > Users > CHARMING > Desktop > kazz > SavingsAccount.java

```java
class SavingsAccount {
    private static double annualInterestRate;
    private double savingsBalance;
    public SavingsAccount(double initialBalance) {
        this.savingsBalance = initialBalance;
    }
    public void calculateMonthlyInterest() {
        double monthlyInterest = (savingsBalance * annualInterestRate) / 12;
        savingsBalance += monthlyInterest;
    }
    public static void modifyInterestRate(double newRate) {
        annualInterestRate = newRate;
    }
    public double getSavingsBalance() {
        return savingsBalance;
    }
    public static void main(String[] args) {
        SavingsAccount saver1 = new SavingsAccount(2000.0);
        SavingsAccount saver2 = new SavingsAccount(3000.0);
        modifyInterestRate(0.04);
        saver1.calculateMonthlyInterest();
        saver2.calculateMonthlyInterest();
        System.out.println("Saver1 balance: $" + saver1.getSavingsBalance());
        System.out.println("Saver2 balance: $" + saver2.getSavingsBalance());
        modifyInterestRate(0.05);
        saver1.calculateMonthlyInterest();
        saver2.calculateMonthlyInterest();
        System.out.println("Saver1 balance: $" + saver1.getSavingsBalance());
        System.out.println("Saver2 balance: $" + saver2.getSavingsBalance());
    }
}
```

OUTPUT



QUESTION 9

Create a class called Book to represent a book. A Book should include four pieces of information as instance variables-a book name, an ISBN number, an author name and a publisher. Your class should have a constructor that initializes the four instance variables. Provide a mutator method and accessor method (query method) for each instance variable. Inaddition, provide a method named getBookInfo that returns the description of the book as a String (the description should include all the information about the book). You should use this keyword in member methods and constructor. Write a test application named BookTest to create an array of object for 30 elements for class Book to demonstrate the class Book's capabilities.

PROGRAM

```java
class Book {
    private String bookName;
    private String isbn;
    private String authorName;
    private String publisher;
    public Book(String bookName, String isbn, String authorName, String publisher) {
        this.bookName = bookName;
        this.isbn = isbn;
        this.authorName = authorName;
        this.publisher = publisher;
    }
    public void setBookName(String bookName) {
        this.bookName = bookName;
    }
    public void setIsbn(String isbn) {
        this.isbn = isbn;
    }
    public void setAuthorName(String authorName) {
        this.authorName = authorName;
    }
    public void setPublisher(String publisher) {
        this.publisher = publisher;
    }
    public String getBookName() {
        return bookName;
    }
    public String getIsbn() {
        return isbn;
    }
    public String getAuthorName() {
        return authorName;
    }
}
```

```java
    public String getAuthorName() {
        return authorName;
    }
    public String getPublisher() {
        return publisher;
    }
    public String getBookInfo() {
        return "Book Name: " + bookName + "\nISBN: " + isbn + "\nAuthor: " + authorName + "\nPublisher: " + publisher;
    }
    public static void main(String[] args) {
        Book book = new Book("Java Programming", "12345", "John Doe", "Tech Publishers");
        System.out.println(book.getBookInfo());
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Book.java
PS C:\Users\CHARMING\desktop\kazz> java Book
Book Name: Java Programming
ISBN: 12345
Author: John Doe
Publisher: Tech Publishers
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 10

. (Complex Class) Create a class called Complex for performing arithmetic with complex numbers. Write a program to test your class. Complex numbers have the form realPart + imaginaryPart * I where i is Use double variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should contain default values in case no initializers are provided. Provide public member functions that perform the following tasks: a. Adding two Complex numbers: The real parts are added together and the imaginary parts are added together. b. Subtracting two Complex numbers: The real part of the right operand is subtracted from the real part of the left operand, and the imaginary part of the right operand is subtracted from the imaginary part of the left operand. c. Printing Complex numbers in the form (a, b), where a is the real part and b is the imaginary part

PROGRAM

```java
class Complex {
    private double realPart;
    private double imaginaryPart;
    public Complex(double realPart, double imaginaryPart) {
        this.realPart = realPart;
        this.imaginaryPart = imaginaryPart;
    }
    public Complex add(Complex other) {
        return new Complex(this.realPart + other.realPart, this.imaginaryPart + other.imaginaryPart);
    }
    public Complex subtract(Complex other) {
        return new Complex(this.realPart - other.realPart, this.imaginaryPart - other.imaginaryPart);
    }
    public void print() {
        System.out.println("(" + realPart + ", " + imaginaryPart + "i)");
    }
    public static void main(String[] args) {
        Complex c1 = new Complex(3.0, 4.5);
        Complex c2 = new Complex(1.5, 2.5);
        Complex sum = c1.add(c2);
        Complex difference = c1.subtract(c2);
        System.out.print("Sum: ");
        sum.print();
        System.out.print("Difference: ");
        difference.print();
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Complex.java
PS C:\Users\CHARMING\desktop\kazz> java Complex
Sum: (4.5, 7.0i)
Difference: (1.5, 2.0i)
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 11

. (Rational Class) Create a class called Rational for performing arithmetic with fractions. Write a program to test your class. Use integer variables to represent the private data of the classthe numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should contain default values in case no initializers are provided and should store the fraction in reduced form. For example, the fraction would be stored in the object as 1 in the numerator and 2 in the denominator. Provide public member functions that perform each of the following tasks: a. Adding two Rational numbers. The result should be stored in reduced form. b. Subtracting two Rational numbers. The result should be stored in reduced form. c. Multiplying two Rational numbers. The result should be stored in reduced form. d. Dividing two Rational numbers. The result should be stored in reduced form. e. Printing Rational numbers in the form a/b, where a is the numerator and b is the denominator.

f. Printing Rational numbers in floating-point format.

PROGRAM

```
class Rational {
    private int numerator;
    private int denominator;
    public Rational(int numerator, int denominator) {
        if (denominator == 0) {
            throw new IllegalArgumentException("Denominator cannot be zero.");
        }
        int gcd = gcd(numerator, denominator);
        this.numerator = numerator / gcd;
        this.denominator = denominator / gcd;
    }
    private int gcd(int a, int b) {
        return b == 0 ? a : gcd(b, a % b);
    }
    public Rational add(Rational other) {
        int num = this.numerator * other.denominator + this.denominator * other.numerator;
        int den = this.denominator * other.denominator;
        return new Rational(num, den);
    }
    public Rational subtract(Rational other) {
        int num = this.numerator * other.denominator - this.denominator * other.numerator;
        int den = this.denominator * other.denominator;
        return new Rational(num, den);
    }
    public Rational multiply(Rational other) {
        return new Rational(this.numerator * other.numerator, this.denominator * other.denominator);
    }
    public Rational divide(Rational other) {
        return new Rational(this.numerator * other.denominator, this.denominator * other.numerator);
    }
```

```
    public void printAsFraction() {
        System.out.println(numerator + "/" + denominator);
    }
    public void printAsDecimal() {
        System.out.println((double) numerator / denominator);
    }
    public static void main(String[] args) {
        Rational r1 = new Rational(1, 2);
        Rational r2 = new Rational(2, 3);
        Rational sum = r1.add(r2);
        Rational difference = r1.subtract(r2);
        Rational product = r1.multiply(r2);
        Rational quotient = r1.divide(r2);
        System.out.print("Sum: ");
        sum.printAsFraction();
        System.out.print("Difference: ");
        difference.printAsFraction();
        System.out.print("Product: ");
        product.printAsFraction();
        System.out.print("Quotient: ");
        quotient.printAsFraction();
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac Rational.java
PS C:\Users\CHARMING\desktop\kazz> java Rational
Sum: 7/6
Difference: 1/-6
Product: 1/3
Quotient: 3/4
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 12

(HugeInteger Class) Create a class HugeInteger that uses a 40-element array of digits to store integers as large as 40 digits each. Provide member functions input, output, add and substract. For comparing HugeInteger objects, provide functions isEqualTo,

isNotEqualTo, isGreaterThan, isLessThan, isGreaterThanOrEqualTo and isLessThanOrEqualToeach of these is a "predicate" function that simply returns TRue if the relationship holds between the two HugeIntegers and returns false if the relationship does not hold. Also, provide a predicate function isZero. If you feel ambitious, provide member functions multiply, divide and modulus.

PROGRAM

```
class HugeInteger {
    private int[] digits = new int[40];
    public HugeInteger(String number) {
        for (int i = 0; i < number.length(); i++) {
            digits[40 - number.length() + i] = Character.getNumericValue(number.charAt(i));
        }
    }
    public HugeInteger add(HugeInteger other) {
        HugeInteger result = new HugeInteger("0");
        int carry = 0;
        for (int i = 39; i >= 0; i--) {
            int sum = this.digits[i] + other.digits[i] + carry;
            result.digits[i] = sum % 10;
            carry = sum / 10;
        }  return result;
    }
    public void print() {
        boolean leadingZero = true;
        for (int digit : digits) {
            if (digit != 0) leadingZero = false;
            if (!leadingZero) System.out.print(digit);
        }
        if (leadingZero) System.out.print("0");
        System.out.println();
    }
    public static void main(String[] args) {
        HugeInteger num1 = new HugeInteger("12345678901234567890");
        HugeInteger num2 = new HugeInteger("98765432109876543210");
        HugeInteger sum = num1.add(num2);
        System.out.print("Sum: ");
        sum.print();
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac HugeInteger.java
PS C:\Users\CHARMING\desktop\kazz> java HugeInteger
Sum: 111111111011111111100
PS C:\Users\CHARMING\desktop\kazz>
```

QUESTION 13

(TicTacToe Class) Create a class TicTacToe that will enable you to write a complete program to play the game of tic-tac-toe. The class contains as private data a 3-by-3 twodimensional array of integers. The constructor should initialize the empty board to all zeros. Allow two human players. Wherever the first player moves, place a 1 in the specified square. Place a 2 wherever the second player moves. Each move must be to an empty square. After each move, determine whether the game has been won or is a draw. If you feel ambitious, modify your program so that the computer makes the moves for one of the players. Also, allow the player to specify whether he or she wants to go first or second. If you feel exceptionally ambitious, develop a program that will play three-dimensional tic-tactoe on a 4-by-4-by-4 board. [Caution: This is an extremely challenging project that could take many weeks of effort!]

PROGRAM

```java
import java.util.Scanner;

class TicTacToe {
    private int[][] board = new int[3][3];
    private int currentPlayer = 1;

    public boolean makeMove(int row, int col) {
        if (row < 0 || row > 2 || col < 0 || col > 2 || board[row][col] != 0) {
            return false;
        }
        board[row][col] = currentPlayer;
        currentPlayer = (currentPlayer == 1) ? 2 : 1;
        return true;
    }
    public int checkWinner() {
        for (int i = 0; i < 3; i++) {
            if (board[i][0] != 0 && board[i][0] == board[i][1] && board[i][1] == board[i][2]) return board[i][0];
            if (board[0][i] != 0 && board[0][i] == board[1][i] && board[1][i] == board[2][i]) return board[0][i];
        }
        if (board[0][0] != 0 && board[0][0] == board[1][1] && board[1][1] == board[2][2]) return board[0][0];
        if (board[0][2] != 0 && board[0][2] == board[1][1] && board[1][1] == board[2][0]) return board[0][2];
        return 0;
    }
    public boolean isDraw() {
        for (int[] row : board) {
            for (int cell : row) {
                if (cell == 0) return false;
            }
        }
        return checkWinner() == 0;
    }
```

```java
    public void printBoard() {
        for (int[] row : board) {
            for (int cell : row) {
                System.out.print((cell == 0 ? "-" : (cell == 1 ? "X" : "O")) + "");
            }
            System.out.println();
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        TicTacToe game = new TicTacToe();

        while (true) {
            game.printBoard();
            System.out.println("Player " + (game.currentPlayer == 1 ? "X" : "O") + ", enter your move (row and column): ");
            int row = scanner.nextInt();
            int col = scanner.nextInt();

            if (!game.makeMove(row, col)) {
                System.out.println("Invalid move! Try again.");
                continue;
            }
            int winner = game.checkWinner();
            if (winner != 0) {
                game.printBoard();
                System.out.println("Player " + (winner == 1 ? "X" : "O") + " wins!");
                break;
            }
```

```
            if (game.isDraw()) {
                game.printBoard();
                System.out.println("It's a draw!");
                break;
            }
        }
        scanner.close();
    }
}
```

OUTPUT

```
PS C:\Users\CHARMING> cd desktop
PS C:\Users\CHARMING\desktop> cd kazz
PS C:\Users\CHARMING\desktop\kazz> javac TicTacToe.java
PS C:\Users\CHARMING\desktop\kazz> java TicTacToe
---
---
---
Player X, enter your move (row and column):
1
1
---
-X-
---
Player O, enter your move (row and column):
1
2
---
-XO
---
Player X, enter your move (row and column):
```