PROBLEMA 1

Clase Operaciones

```
public class MateService {

  // Para los numeros Perfectos
  public String CalculatePerfect(int num1) {
    int num2 = 0;
    for (int i = 1; i <= num1 / 2; i++) {
      if (num1 % i == 0) {
        num2 += i;
      }
    }
    return (num2 == num1) ? "Si es un numero perfecto"
      : "No es un numero perfecto";
  }

  // Para los numeros amigos
  public String CalculateAmigos(int n1, int n2) {
    int aux1 = sumaDigitos(n1);
    int aux2 = sumaDigitos(n2);
    String rpta = (aux1 == n2 && aux2 == n1) ? "Son numeros amigos"
        : "No son amigos";
    return rpta;
  }

  private int sumaDigitos(int num) {
    int suma = 0;
    for (int i = 1; i <= num / 2; i++) {
      suma += (num % i == 0) ? i : 0;
    }
    return suma;
  }

  // Para la serie
  public double CalculateSerie(int n, double x) {
    double result = 0;
    for (int i = 0; i <= n; i++) {
      result += (Math.pow(-1, i) * Math.pow(x, 2 * i + 1)) / (2 * i + 1);
    }
    return result;
  }

}
```

Prueba de Numero Perfecto

```
public class PruebaPerfecto {

    public static void main(String[] args) {
        //Creacion
        MateService test = new MateService();
        //Verificacion de funciones
        //Modifique el 28
        System.out.println(test.CalculatePerfect(8));
    }
}

Si es un numero perfecto
```

Prueba de Números Amigos

```
public class PruebaAmigos {

    public static void main(String[] args) {
        MateService test = new MateService();
        System.out.println(test.CalculateAmigos(220, 284));
    }
}

Son numeros amigos
```

Prueba de la Serie

```
public class PruebaSerie {

   public static void main(String[] args) {
     //Creacion
     MateService test = new MateService();
     //Verificacion de funciones
     double x = 1;
     System.out.println("\tx\tn\tserie");
     for (int n = 0; n <= 5; n++) {
       double s = test.CalculateSerie(n, x);
       System.out.println("\t" + x + "\t" + n + "\t" + s);
     }
   }
}
```

```
x  n  serie
1.0  0  1.0
1.0  1  0.6666666666666667
1.0  2  0.8666666666666667
1.0  3  0.7238095238095239
1.0  4  0.8349206349206351
1.0  5  0.7440115440115441
```

PROBLEMA 2

Clase VectorService

```
import java.util.Random;

public class VectorService {

  private int[] vector3;
  private int[] vector5;
  private Random random = new Random();

  public VectorService(int n) {
    this.vector3 = new int[n];
    this.vector5 = new int[n];
    for (int i = 0; i < n; i++) {
      this.vector3[i] = (random.nextInt(13) + 4) * 3;
      this.vector5[i] = (random.nextInt(8) + 2) * 5;
    }
  }

  public int[] getVector3() {
    return this.vector3;
  }

  public int[] getVector5() {
    return this.vector5;
  }

  public int[] vectorComun() {
    int[] aux = new int[getVector3().length];
    boolean repeat = false;
    for (int i = 0; i < getVector3().length; i++) {
      for (int j = 0; j < getVector5().length; j++) {
        if (getVector3()[i] == getVector5()[j]) {
          for (int k : aux) {
            if (k == getVector5()[j]) {
```

```
                repeat = true;
                break;
              }
            }
            if (repeat == false) {
              aux[j] = getVector3()[i];
            }
            repeat = false;
          }
        }
      }
      int count = 0;
      for (int i = 0; i < aux.length; i++) {
        if (aux[i] != 0) {
          count++;
        }
      }
      int[] rspt = new int[count];
      count = 0;
      for (int i = 0; i < aux.length; i++) {
        if (aux[i] != 0) {
          rspt[count] = aux[i];
          count++;
        }
      }
      return rspt;
    }

    public int[] vectorUnion() {
      int[] aux = new int[getVector3().length * 2];
      boolean repeat = false;
      int count = 0;
      for (int i = 0; i < getVector3().length; i++) {
        for (int j : aux) {
          if (j == getVector3()[i]) {
            repeat = true;
            break;
          }
        }
        if (repeat == false) {
          aux[count] = getVector3()[i];
          count++;
        }
        repeat = false;
      }
      for (int i = 0; i < getVector5().length; i++) {
        for (int j : aux) {
```

```
          if (j == getVector5()[i]) {
            repeat = true;
            break;
          }
        }
        if (repeat == false) {
          aux[count] = getVector5()[i];
          count++;
        }
        repeat = false;
      }
      count = 0;
      for (int i = 0; i < aux.length; i++) {
        if (aux[i] != 0) {
          count++;
        }
      }
      int[] rspt = new int[count];
      count = 0;
      for (int i = 0; i < aux.length; i++) {
        if (aux[i] != 0) {
          rspt[count] = aux[i];
          count++;
        }
      }
      return rspt;
    }

}
```

Clase de Prueba

```
public class Pruebavectores {

  public static void main(String[] args) {
    // Modifique el "n" del constructor
    VectorService test = new VectorService(5);
    // Primer Vector
    System.out.println("Primer vector: " +
      Arrays.toString(test.getVector3()));
    // Segundo Vector
    System.out.println("Segundo vector: " +
      Arrays.toString(test.getVector5()));
    // Vector Union
    System.out.println("Vector Union: " +
```

```
        Arrays.toString(test.vectorUnion()));
      // Vector Comun
      System.out.println("Vector Comun: " +
        Arrays.toString(test.vectorComun()));
    }
}

Primer vector: [12, 21, 27, 18, 15]
Segundo vector: [10, 15, 35, 15, 15]
Vector Union: [12, 21, 27, 18, 15, 10, 35]
Vector Comun: [15]
```