

Laboratorio 2 INF-121

Nombre: Ticona Tapia Jhonatan Gerardo

CI:13971678

Fecha: 10/09/25

1. Algebra Vectorial. Se quiere construir un programa orientado a objetos con sobrecarga de funciones. Para la solución del problema se debe dibujar el diagrama de clases, acompañado del respectivo código de solución. El programa debe: (1) determinar si dos vectores son perpendiculares, (2) determinar si dos vectores son paralelos, (3) determinar la proyección de dos vectores, (4) determinar el componente de dos vectores. Para ello genere la clase AlgebraVectorial con la sobrecarga de constructores y sobrecargando los siguientes métodos:

Código:

```
package ejercicio1;
public class AlgebraVectorial {
    private double x;
    private double y;
    private double z;

    public AlgebraVectorial() {
        x=0;
        y=0;
        z=0;
    }

    public AlgebraVectorial(double x, double y) {
        this.x=x;
        this.y=y;
        z=0;
    }

    public AlgebraVectorial(double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }

    public boolean perpendicular(AlgebraVectorial b,double y2,double z2) {
        AlgebraVectorial suma = new AlgebraVectorial(this.x + b.x, this.y +y2, this.z + z2);
        AlgebraVectorial resta = new AlgebraVectorial(this.x - b.x, this.y - y2, this.z - z2);
        return Math.abs(suma.norma() - resta.norma()) < 1e-9;
    }

    public boolean perpendicular(double x2,double y2,double z2) {
        AlgebraVectorial resta1 = new AlgebraVectorial(this.x - x2, this.y - y2, this.z - z2);
        AlgebraVectorial resta2 = new AlgebraVectorial(x2 - this.x, y2 - this.y, z2- this.z);
        return Math.abs(resta1.norma() - resta2.norma()) < 1e-9;
    }

    public boolean perpendicular(AlgebraVectorial b) {
        return Math.abs(this.productoPunto(b)) < 1e-9;
    }

    public boolean perpendicular(AlgebraVectorial b,double y2) {
        AlgebraVectorial suma = new AlgebraVectorial(this.x + b.x, this.y + y2, this.z + b.z);
        double izquierda = Math.pow(suma.norma(), 2);
        double derecha = Math.pow(this.norma(), 2) + Math.pow(b.norma(), 2);
        return Math.abs(izquierda - derecha) < 1e-9;
    }

    public boolean paralela(AlgebraVectorial b) {
        AlgebraVectorial cruz = this.productoCruz(b);
        return Math.abs(cruz.x) < 1e-9 && Math.abs(cruz.y) < 1e-9 && Math.abs(cruz.z) < 1e-9;
    }
}
```

```

public boolean paralela1(AlgebraVectorial b) {
    AlgebraVectorial cruz = this.productoCruz(b);
    return Math.abs(cruz.x) < 1e-9 && Math.abs(cruz.y) < 1e-9 && Math.abs(cruz.z) < 1e-9;
}

public AlgebraVectorial Proyeccion_de_a_sobre_b(AlgebraVectorial b) {
    double escalar = this.productoPunto(b) / Math.pow(b.norma(), 2);
    return new AlgebraVectorial(escalar * b.x, escalar * b.y, escalar * b.z);
}

public double Componente_de_a_en_b(AlgebraVectorial b) {
    return this.productoPunto(b) / b.norma();
}

@Override
public String toString() {
    return String.format("x: %.2f y: %.2f z: %.2f", x, y, z);
}

public double productoPunto(AlgebraVectorial v) {
    return this.x * v.x + this.y * v.y + this.z * v.z;
}

public AlgebraVectorial productoCruz(AlgebraVectorial v) {
    double cx = this.y * v.z - this.z * v.y;
    double cy = this.z * v.x - this.x * v.z;
    double cz = this.x * v.y - this.y * v.x;
    return new AlgebraVectorial(cx, cy, cz);
}

public double norma() {
    return Math.sqrt(x * x + y * y + z * z);
}

public double getX() {
    return x;
}

public void setX(double x) {
    this.x = x;
}

public double getY() {
    return y;
}

public void setY(double y) {
    this.y = y;
}

public double getZ() {
    return z;
}

```

AlgebraVectorial

- x double

- y double

- z double

+AlgebraVectorial()

+AlgebraVectorial(double x, double y)

+AlgebraVectorial(double x, double y, double z)

+ perpendicular(AlgebraVectorial b, double y2, double z2) :boolean

+ perpendicular(double x2, double y2, double z2) :boolean

+ perpendicular(AlgebraVectorial b) :boolean

+ perpendicular(AlgebraVectorial b, double y2) :boolean

+ paralela(AlgebraVectorial b) :boolean

+ paralela1(AlgebraVectorial b) :boolean

+Proyeccion_de_a_sobre_b(AlgebraVectorial b) :AlgebraVectorial

+Componente_de_a_en_b(AlgebraVectorial b) :double

+ toString() :String

+ productoPunto(AlgebraVectorial v) :double

+ productoCruz(AlgebraVectorial v) :AlgebraVectorial

+ norma() :double

+ Getters and Setters

```

package ejercicio1;
public class Tests {

    public static void main(String[] args) {
        AlgebraVectorial a = new AlgebraVectorial(2, 4, 5);
        AlgebraVectorial b = new AlgebraVectorial(-3, 7, 4);

        System.out.println("a = " + a);
        System.out.println("b = " + b);

        System.out.println("a) |a+b| = |a-b|?: " + a.perpendicular(b, b.getY(), b.getZ()));
        System.out.println("b) |a-b| = |b-a|?: " + a.perpendicular(b, b.getX(), b.getZ()));
        System.out.println("c) a*b = 0?: " + a.perpendicular(b));
        System.out.println("d) |a+b|^2 = |a|^2 + |b|^2?: " + a.perpendicular(b, b.getY()));
        System.out.println("e) a = r*b?: " + a.paralela(b));
        System.out.println("f) a x b = 0?: " + a.paralela1(b));
        System.out.println("g) Proyeccion de a sobre b: " + a.Proyeccion_de_a_sobre_b(b));
        System.out.println("h) Componente de a en direccion de b: " + a.Componente_de_a_en_b(b));
    }
}

```

Resultado:

```
--- exec:3.1.0:exec (default-cli) @ Practica2Inf121 ---
a = x: 2,00 y: 4,00 z: 5,00
b = x: -3,00 y: 7,00 z: 4,00
a) |a+b| = |a-b|?: false
b) |a-b| = |b-a|?: true
c) a*b = 0?: false
d) |a+b|^2 = |a|^2 + |b|^2?: false
e) a = r*b?: false
f) a x b = 0?: false
g) Proyeccion de a sobre b: x: -1,70 y: 3,97 z: 2,27
h) Componente de a en direccion de b: 4.88240082724041
-----
BUILD SUCCESS
```

2. En algebra vectorial un vector tridimensional se define como: $a = (a_1 \ a_2 \ a_3)$. Además se tiene lo siguiente:

Código:

```
package ejercicio2;
public class Vector3D {
    private double x, y, z;

    public Vector3D() {
        x = 0;
        y = 0;
        z = 0;
    }

    public Vector3D(double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }

    public Vector3D add(Vector3D v) {
        return new Vector3D(this.x + v.x, this.y + v.y, this.z + v.z);
    }

    public Vector3D add(double r) {
        return new Vector3D(r * this.x, r * this.y, r * this.z);
    }

    public double add() {
        return Math.sqrt(x * x + y * y + z * z);
    }

    public Vector3D scale() {
        double mag = this.add();
        return this.add(1.0 / mag);
    }

    public double dot(Vector3D v) {
        return this.x * v.x + this.y * v.y + this.z * v.z;
    }

    public Vector3D cross(Vector3D v) {
        return new Vector3D(
            this.y * v.z - this.z * v.y,
            this.z * v.x - this.x * v.z,
            this.x * v.y - this.y * v.x
        );
    }

    public Vector3D projectOnto(Vector3D v) {
        double scalarProjection = this.dot(v) / v.add();
        return v.scale().add(scalarProjection);
    }

    public boolean isOrthogonalTo(Vector3D v) {
        return Math.abs(this.dot(v)) < 1e-10;
    }

    @Override
    public String toString() {
        return String.format("%.2f, %.2f, %.2f", x, y, z);
    }
}
```

Vector3D	
- x double	
- y double	
- z double	
<hr/>	
+Vector3D()	
+Vector3D(double x, double y, double z)	
+add(Vector3D v)	:Vector3D
+add(double r)	:Vector3D
+add()	:double
+scale()	:Vector3D
+dot(Vector3D v)	:double
+cross(Vector3D v)	:Vector3D
+projectOnto(Vector3D v)	:Vector3D
+isOrthogonalTo(Vector3D v)	:boolean
+toString()	:String

```

package ejercicio2;
public class Test {
    public static void main(String[] args) {
        Vector3D a = new Vector3D(1, 2, 3);
        Vector3D b = new Vector3D(4, 5, 6);

        System.out.println("Vector a: " + a);
        System.out.println("Vector b: " + b);

        System.out.println("Suma: a + b = " + a.add(b));
        System.out.println("Escalar * a (2.0): " + a.add(2.0));
        System.out.println("Longitud de a: " + a.add());
        System.out.println("Normalizado de a: " + a.scale());
        System.out.println("Producto escalar a . b: " + a.dot(b));
        System.out.println("Producto vectorial a x b: " + a.cross(b));
        System.out.println("Proyeccion de a sobre b: " + a.projectOnto(b));
        System.out.print("a es perpendicular u ortogonal a b?: ");
        if(a.isOrthogonalTo(b)){
            System.out.println("Si");
        }
        else{
            System.out.println("no");
        }
    }
}

```

Resultado:

```

--- exec:3.1.0:exec (default-cli) @ Practica2Inf121 ---
Vector a: (1,00, 2,00, 3,00)
Vector b: (4,00, 5,00, 6,00)
Suma: a + b = (5,00, 7,00, 9,00)
Escalar * a (2.0): (2,00, 4,00, 6,00)
Longitud de a: 3.7416573867739413
Normalizado de a: (0,27, 0,53, 0,80)
Producto escalar a . b: 32.0
Producto vectorial a x b: (-3,00, 6,00, -3,00)
Proyeccion de a sobre b: (1,66, 2,08, 2,49)
a es perpendicular u ortogonal a b?: No
-----
BUILD SUCCESS
-----

```