

## Universidad Nacional Autónoma de México



## Facultad de Estudios Superiores Aragón

## Ingeniería en Computación

Estructura de Datos

Jesús Hernández Cabrera

**Axel Yahir Moreno Rodríguez** 

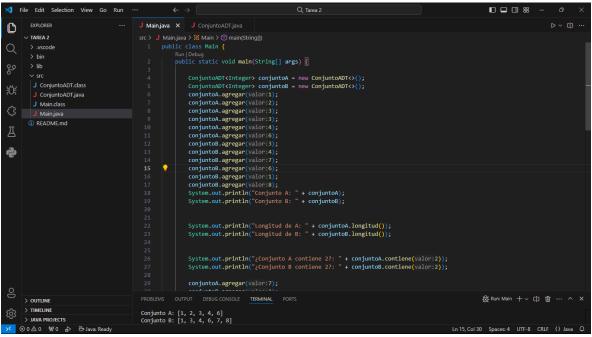


Turno Vespertino Grupo 1360

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
                                                                                                                                                                                                                                                                                                                                                                    袋 Run: Main + ∨ □ 前 ··· ^ ×
    Unión de A y B: [2, 3, 4, 6, 7, 8]
Intersección de A y B: [2, 3, 4, 6, 7]
Diferencia de B y A (B-A): []
PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2> ^C
    PS C:\Users\axeLx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2>
C:; cd 'c:\Users\axeLx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2\Tarea
     Longitud de B: 6
     ¿Conjunto A contiene 2?: true
      ¿Conjunto B contiene 2?: false
    Yeliminamos 1 al Conjunto A: [2, 3, 4, 6, 7] Si agregamos 2 y eliminamos 1 al Conjunto B: [2, 3, 4, 6, 7, 8] Si agregamos 7 y eliminamos 1 al Conjunto B: [2, 3, 4, 6, 7, 8] A es igual a B?: false A es subconjunto de B?: false
    Unión de A y B: [2, 3, 4, 6, 7, 8]

Intersección de A y B: [2, 3, 4, 6, 7]

Diferencia de B y A (B-A): [8]
     PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2>
    PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
                                                                                                                                                                                                                                                                                                                                                                    以 Run: Main + ∨ □ 前 ··· ^ ×
   PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2> ^{\circ} PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2>
    PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\ c:; cd 'c:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2\Tarea 2'; & 'c:\Program Files\Java\jre-1.8\bin\java.exe' '-cp' 'C:\Users\axelx\Documents\FES\3er semestre\Estructura
    tura de Datos\Tareas\Tarea 2\Tarea 2'; & 'C:\Program Files\Java\jre-1.8\bin\java.exe'
de Datos\Tareas\Tarea 2\Tarea 2\bin' 'Main'
    Conjunto A: [1, 2, 3, 4, 6]
Conjunto B: [1, 3, 4, 6, 7, 8]
Longitud de A: 5
     Longitud de B: 6
    ¿Conjunto A contiene 2?: true ¿Conjunto B contiene 2?: false Si agregamos 2 y eliminamos 1 al Conjunto A: [2, 3, 4, 6, 7] Si agregamos 7 y eliminamos 1 al Conjunto B: [2, 3, 4, 6, 7, 8] ¿A es igual a B?: false ¿A es subconjunto de B?: false
    Unión de A y B: [2, 3, 4, 6, 7, 8]
Intersección de A y B: [2, 3, 4, 6, 7]
Diferencia de A y B (A-B): []
    PS C:\Users\axelx\Documents\FES\3er semestre\Estructura de Datos\Tareas\Tarea 2\Tarea 2>
📢 File Edit Selection View Go Run …
```



```
public class Main {
    public static void main(String[] args) {
        ConjuntoADT<Integer> conjuntoA = new ConjuntoADT<>();
        ConjuntoADT<Integer> conjuntoB = new ConjuntoADT<>();
        conjuntoA.agregar(1);
        conjuntoA.agregar(2);
        conjuntoA.agregar(3);
        conjuntoA.agregar(3);
        conjuntoA.agregar(4);
        conjuntoA.agregar(6);
        conjuntoB.agregar(3);
        conjuntoB.agregar(4);
        conjuntoB.agregar(7);
        conjuntoB.agregar(6);
        conjuntoB.agregar(1);
        conjuntoB.agregar(8);
        System.out.println("Conjunto A: " + conjuntoA);
        System.out.println("Conjunto B: " + conjuntoB);
        System.out.println("Longitud de A: " + conjuntoA.longitud());
        System.out.println("Longitud de B: " + conjuntoB.longitud());
        System.out.println("¿Conjunto A contiene 2?: " +
conjuntoA.contiene(2));
        System.out.println("¿Conjunto B contiene 2?: " +
conjuntoB.contiene(2));
        conjuntoA.agregar(7);
        conjuntoB.agregar(2);
        conjuntoA.eliminar(1);
        conjuntoB.eliminar(1);
        System.out.println("Si agregamos 2 y eliminamos 1 al Conjunto A: " +
conjuntoA);
        System.out.println("Si agregamos 7 y eliminamos 1 al Conjunto B: " +
conjuntoB);
        System.out.println("¿A es igual a B?: " +
conjuntoA.equals(conjuntoB));
        System.out.println("¿A es subconjunto de B?: " +
conjuntoA.esSubConjunto(conjuntoB));
```

```
ConjuntoADT<Integer> conjuntoUnion = conjuntoA.union(conjuntoB);
        System.out.println("Unión de A y B: " + conjuntoUnion);
        ConjuntoADT<Integer> conjuntoInterseccion =
conjuntoA.interseccion(conjuntoB);
        System.out.println("Intersección de A y B: " +
conjuntoInterseccion);
        ConjuntoADT<Integer> conjuntoDiferencia =
conjuntoA.diferencia(conjuntoB);
        System.out.println("Diferencia de A y B (A-B): " +
conjuntoDiferencia);
import java.util.HashSet;
import java.util.Set;
public class ConjuntoADT<T> {
    private Set<T> conjunto;
    public ConjuntoADT() {
        this.conjunto = new HashSet<>();
    public int longitud() {
        return conjunto.size();
    public boolean contiene(T valor) {
        return conjunto.contains(valor);
    public void agregar(T valor) {
        conjunto.add(valor);
    public void eliminar(T valor) {
        conjunto.remove(valor);
```

```
public boolean equals(ConjuntoADT<T> otroConjunto) {
    return this.conjunto.equals(otroConjunto.conjunto);
public boolean esSubConjunto(ConjuntoADT<T> otroConjunto) {
    return this.conjunto.containsAll(otroConjunto.conjunto);
public ConjuntoADT<T> union(ConjuntoADT<T> otroConjunto) {
    ConjuntoADT<T> nuevoConjunto = new ConjuntoADT<>();
    nuevoConjunto.conjunto.addAll(this.conjunto);
    nuevoConjunto.conjunto.addAll(otroConjunto.conjunto);
   return nuevoConjunto;
public ConjuntoADT<T> interseccion(ConjuntoADT<T> otroConjunto) {
    ConjuntoADT<T> nuevoConjunto = new ConjuntoADT<>();
    for (T valor : this.conjunto) {
        if (otroConjunto.contiene(valor)) {
            nuevoConjunto.agregar(valor);
   return nuevoConjunto;
public ConjuntoADT<T> diferencia(ConjuntoADT<T> otroConjunto) {
    ConjuntoADT<T> nuevoConjunto = new ConjuntoADT<>();
    for (T valor : this.conjunto) {
        if (!otroConjunto.contiene(valor)) {
            nuevoConjunto.agregar(valor);
   return nuevoConjunto;
@Override
public String toString() {
   return conjunto.toString();
```