

LAB – 121
LABORATORIO DE INF – 121
“POLIMORFISMO
(SOBRECARGA DE METODOS Y OPERADORES)”
Gestión II/2016

Docente: Lic. Marcelo Aruquipa
Aux. Pamela Choque
Aux. Fabio Laura

PARTE I POLIMORFISMO

El polimorfismo, como su mismo nombre sugiere múltiples formas, se refiere a la posibilidad de acceder a un variado rango de funciones distintas a través del mismo interfaz. Es decir un mismo identificador puede tener distintas formas (distintos cuerpos de función, distintos comportamientos) dependiendo, en general, del contexto en el que se halle inserto. El polimorfismo se puede establecer mediante la sobrecarga. Sobrecarga.

El polimorfismo es útil cuando se tiene que hacer varios métodos que hacen exactamente lo mismo salvo porque se les pasa un parámetro diferente (o uno más, o uno menos...). No tiene mucho sentido ponerse a llamar al método con nombres diferentes cuando en realidad hacen lo mismo. Por tanto es mucho más lógico y coherente sobrecargar el método. La sobrecarga sirve también para sobrescribir un método creado por la clase padre en el caso de herencia. En este caso no tienes más remedio que llamarle igual si quieres sobrescribirlo.

1. SOBRECARGA DE METODOS

La sobrecarga de métodos consiste en definir más de un método con el mismo nombre, al igual que los constructores.

Estos métodos se diferenciarán por la cantidad y tipos de parámetros con los cuales se definieron.

Sintaxis:

```
// método que no devuelve
tipoAcceso void nomMetodoA(){
    ...
}
tipoAcceso void nomMetodoA(tipoDato p1, tipoDato p2, ...){
    ...
}
// método que devuelve
tipoAcceso tipoDev nomMetodoB(){
    ...
    return valor;
}

tipoAcceso tipoDev nomMetodoB(tipoDato p1, tipoDato p2, ...){
    ...
    return valor;
}
```

La sobrecarga de métodos puede ser implementada en JAVA , CSharp C# y C++

2. SOBRECARGA DE OPERADORES

La sobrecarga de operadores es la capacidad de transformar los operadores de tal manera que este puede aceptar otro tipo de operandos.

La finalidad de sobrecargar un operador es para ampliar la funcionalidad natural de un operador, esto hace que el código sea legible y fácil de entender.

La sobrecarga de operadores ya era posible en C++ , Java no lo incorpora , por lo que C# tiene una ventaja con respecto a JAVA.

2.1 TIPOS DE OPERADORES

Existen dos tipos de operadores:

a. Operadores Unario: admiten un solo operando, como ser:

+, -, !, ~, ++, --

b. Operadores Binarios: admiten dos operandos, como ser:

+, -, *, /, %, &, |, ^, <<, >>, ==, !=, <, >, <=, >=

Obs. En caso de los operadores de comparación, se exige que si se sobrecarga un operador, también hay que sobrecargar su opuesto.

Para sobrecargar un operador hay que definir e implementar un método y asociarlo al operador que se desea sobrecargar.

Sintaxis:

```
//Operador Unario --, ++, !
tipoAcceso static tipoDev operator Operador(tipoDato p) {
    ...
    ...
    return valor;
}

//Operador Binario -, +, *, /, % >, <, >=, <=, !=
tipoAcceso static tipoDev operator Operador(tipoDato p1, tipoDato p2) {
    ...
    ...
    return valor;
}
```

OBSERVACIONES.

- El método definido, es un método que debe de retornar obligatoriamente un valor.
- Uno de los parámetros del método debe ser un objeto de la clase en la que se está implementando el método.
- La sobrecarga de operadores no modifica la jerarquía de operadores.
- Los siguientes operadores **NO** se pueden **SOBRECARGAR**: &&, ||, [], (), =, ., ?, -, new, is, sizeof, typeof .

3. EJEMPLO SOBRECARGA DE METODOS JAVA Y CSharp C#

Sea la clase teatro con atributos de nombre, (nv) número de asientos vendidos y una matriz de los asientos ya vendidos av [nombre-código de asiento-precio]. Sabiendo que los últimos asientos cuestan 150 Bs. , los asientos del medio 250Bs. y los de adelante 500 Bs.

Diagrama de clases

Teatro
- nombre : string
- nv : int
- av [100,3] : string
+ Teatro();
+ Buscar();
+ Buscar(string nom);
+ Vender();
+ Vender(int n);
+ Mostrar();

Resolver

- Crear el método Buscar() para que este busque en la matriz y muestre que personas pagaron para estar adelante.
- Sobrecargar el método Buscar() enviando como parámetro el nombre de una persona para ver si esta compro una entrada para el teatro, de ser así mostrar cuanto pago y el código de su asiento.
- Sobrecargar el método Vender() uno para insertar a una persona a la matriz de asientos vendidos, y el otro enviando como parámetro el número de asientos a vender.

Solucion.

Teatro.cs	Teatro.java
<pre> public class Teatro { private string nombre; private int nv; private string [,] av = new string[100,4]; //CONSTRUCTOR POR DEFECTO public Teatro(){ this.nombre = "16 de Julio"; this.nv = 8; this.av[1,1]="Pedro Perez"; this.av[1,2]="A-20";this.av[1,3]="500"; this.av[2,1]="Juana Alarcon"; this.av[2,2]="C-85";this.av[2,3]="150"; this.av[3,1]="Jaime Quiroz"; this.av[3,2]="C-100";this.av[3,3]="150"; this.av[4,1]="Camila Prado"; this.av[4,2]="B-39";this.av[4,3]="250"; this.av[5,1]="Luis Choque"; this.av[5,2]="B-45";this.av[5,3]="250"; this.av[6,1]="Ivan Condori"; this.av[6,2]="A-25";this.av[6,3]="500"; this.av[7,1]="Andrea Juan"; this.av[7,2]="C-120";this.av[7,3]="150"; this.av[8,1]="Jhoselin Lopez"; this.av[8,2]="C-110";this.av[8,3]="150"; } public void Buscar(){ for(int i=1 ; i <= this.nv ; i++) { </pre>	<pre> public class Teatro { private String nombre; private int nv; private String [][] av = new String[100][4]; //CONSTRUCTOR POR DEFECTO public Teatro(){ this.nombre = "16 de Julio"; this.nv = 8; this.av[1][1]="PedroPerez"; this.av[1][2]="A-20";this.av[1][3]="500"; this.av[2][1]="JuanaAlarcon"; this.av[2][2]="C-85";this.av[2][3]="150"; this.av[3][1]="Jaime Quiroz"; this.av[3][2]="C-98";this.av[3][3]="150"; this.av[4][1]="Camila Prado"; this.av[4][2]="B-39";this.av[4][3]="250"; this.av[5][1]="Luis Choque"; this.av[5][2]="B-45";this.av[5][3]="250"; this.av[6][1]="Ivan Condori"; this.av[6][2]="A-25";this.av[6][3]="500"; this.av[7][1]="Andrea Juan"; this.av[7][2]="C-79";this.av[7][3]="150"; this.av[8][1]="Jhoselin Lopez"; this.av[8][2]="C-84";this.av[8][3]="150"; } public void buscar(){ for(int i=1 ; i <= this.nv ; i++) { int p = Integer.parseInt(this.av[i][3]); </pre>

```

        int p = int.Parse(this.av[i,3]);
        if( p == 500)
            Console.WriteLine("\nNombre: {0} - Codigo de
            asiento {1}",this.av[i,1], this.av[i,2]);
    }
}
public void Buscar(string nom){
    bool sw = false;
    for(int i=1 ; i <= this.nv ; i++){
        if( this.av[i,1] == nom){
            sw = true ;
            Console.WriteLine("La persona {0} si se
            encuentra en el teatro",this.av[i,1]);
            Console.WriteLine("\nCodigo de asiento {0}
            - Precio {1}",this.av[i,2],this.av[i,3]);
            break;
        }
    }
    if(!sw)
        Console.WriteLine("La persona {0} no se
        encuentra en el teatro",nom);
}
public void Vender(){
    Console.WriteLine("\nIntroduzca nombre de la
    persona su asiento y el precio de este.");
    this.nv++;
    this.av[this.nv,1] = Console.ReadLine();
    this.av[this.nv,2] = Console.ReadLine();
    this.av[this.nv,3] = Console.ReadLine();
}
public void Vender(int n){
    int p = this.nv+1;
    this.nv = this.nv+n;
    for(int i=p ; i<=this.nv ; i++){
        Console.WriteLine("\nIntroduzca nombre de la
        persona su asiento y el precio de este.");
        this.av[i,1] = Console.ReadLine();
        this.av[i,2] = Console.ReadLine();
        this.av[i,3] = Console.ReadLine();
    }
}
public void Mostrar()
{
    Console.WriteLine("\nNombre del teatro {0}
    numero de entradas vendidas{1}",
    this.nombre,this.nv);
    for(int i=1 ; i <= this.nv ; i++)
    {
        Console.WriteLine("\n{0} Nombre {1} - Codigo
        de asiento {2} - Precio
        {3}",i,this.av[i,1],
        this.av[i,2],this.av[i,3]);
    }
}
}

```

Program.cs

```

public class Program{
    public static void Main(){
        Teatro t1 = new Teatro();
        //MOSTRANDO ANTES DE HACER CAMBIOS
        t1.Mostrar();
        Console.WriteLine("\n**Buscar asientos caros vendidos");
        t1.Buscar();
        Console.WriteLine("\n\n**Buscar persona x");
        t1.Buscar("Pedro Perez");
        Console.WriteLine("\n\n**Vender asiento para 1 persona");
        t1.Vender();
        Console.WriteLine("\n\n**Vender asientos para npersonas");
        Console.WriteLine("\n¿A cuantas personas desea
        vender entradas?");
        int n = int.Parse(Console.ReadLine());
        t1.Vender(n);
    }
}

```

```

        if( p == 500)
            System.out.println("Nombre:"+this.av[i][1]+"-
            Codigo de asiento "+this.av[i][2]);
    }
}
public void buscar(String nom){
    boolean sw = false;
    for(int i=1 ; i <= this.nv ; i++){
        if( this.av[i][1] == nom){
            sw = true ;
            System.out.println("La persona si se
            encuentra en el teatro sus datos
            son:"+this.av[i][1]);
            System.out.println("Codigo de asiento"
            +this.av[i][2] + "Precio"+this.av[i][3]);
            break;
        }
    }
    if(!sw)
        System.out.println("La persona "+nom+" no se
        encuentra en el teatro");
}
public void vender(){
    Scanner lee = new Scanner(System.in);
    System.out.println("Introduzca nombre de la
    persona su asiento y el precio de este.");
    this.nv++;
    this.av[this.nv][1] = lee.nextLine();
    this.av[this.nv][2] = lee.nextLine();
    this.av[this.nv][3] = lee.nextLine();
}
public void vender(int n){
    Scanner lee = new Scanner(System.in);
    int p = this.nv+1;
    this.nv = this.nv+n;
    for(int i=p ; i<=this.nv ; i++){
        System.out.println("Introduzca nombre de la
        persona su asiento y el precio de este.");
        this.av[i][1] = lee.nextLine();
        this.av[i][2] = lee.nextLine();
        this.av[i][3] = lee.nextLine();
    }
}
public void mostrar()
{
    System.out.println("Nombre del teatro "
    +this.nombre+" numero de entradas vendidas "+
    this.nv);
    for(int i=1 ; i <= this.nv ; i++)
        System.out.println(i+" Nombre"+
        this.av[i][1]+" - Codigo de asiento "+
        this.av[i][2]+" - Precio "+this.av[i][3]);
}
}

```

Main.java

```

public class Main{
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner lee = new Scanner(System.in);
        Teatro t1 = new Teatro();
        //MOSTRANDO ANTES DE HACER CAMBIOS
        t1.mostrar();
        System.out.println("***Buscar asientos caros vendidos");
        t1.buscar();
        System.out.println("***Buscar persona x");
        t1.buscar("Pedro Perez");
        System.out.println(" Vender asiento para 1 persona");
        t1.vender();
        System.out.println("***Vender asientos para npersonas");
        System.out.println("¿A cuantas personas desea vender
        entradas?");
    }
}

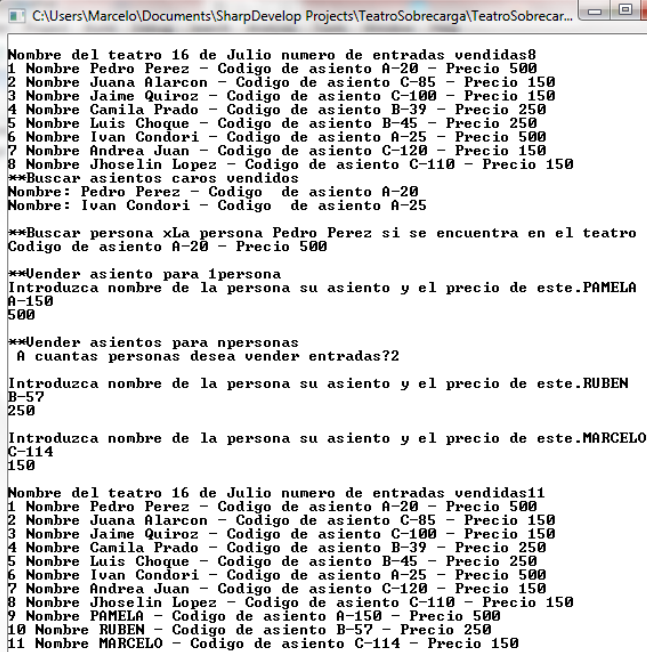
```

```
//MOSTRANDO DESPUES DE LOS CAMBIO
t1.Mostrar();
Console.ReadKey();
}
}
```

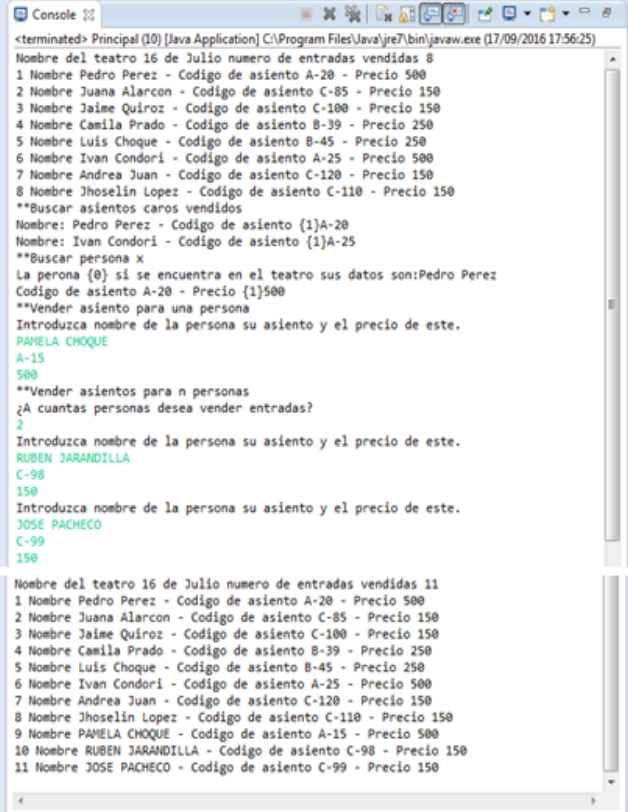
```
int n = lee.nextInt();
t1.vender(n);
//MOSTRANDO DESPUES DE LOS CAMBIOS
t1.mostrar();
}
}
```

Captura de pantalla cs

Captura de pantalla java



```
Nombre del teatro 16 de Julio numero de entradas vendidas8
1 Nombre Pedro Perez - Codigo de asiento A-20 - Precio 500
2 Nombre Juana Alarcon - Codigo de asiento C-85 - Precio 150
3 Nombre Jaime Quiroz - Codigo de asiento C-100 - Precio 150
4 Nombre Camila Prado - Codigo de asiento B-39 - Precio 250
5 Nombre Luis Choque - Codigo de asiento B-45 - Precio 250
6 Nombre Ivan Condori - Codigo de asiento A-25 - Precio 500
7 Nombre Andrea Juan - Codigo de asiento C-120 - Precio 150
8 Nombre Jhoselin Lopez - Codigo de asiento C-110 - Precio 150
**Buscar asientos caros vendidos
Nombre: Pedro Perez - Codigo de asiento A-20
Nombre: Ivan Condori - Codigo de asiento A-25
**Buscar persona x
Nombre: Pedro Perez - Codigo de asiento {1}A-20
Nombre: Ivan Condori - Codigo de asiento {1}A-25
**Vender asiento para una persona
Introduzca nombre de la persona su asiento y el precio de este.PAMELA
A-150
500
**Vender asientos para n personas
A cuantas personas desea vender entradas?2
Introduzca nombre de la persona su asiento y el precio de este.RUBEN
B-57
250
Introduzca nombre de la persona su asiento y el precio de este.MARCELO
C-114
150
Nombre del teatro 16 de Julio numero de entradas vendidas11
1 Nombre Pedro Perez - Codigo de asiento A-20 - Precio 500
2 Nombre Juana Alarcon - Codigo de asiento C-85 - Precio 150
3 Nombre Jaime Quiroz - Codigo de asiento C-100 - Precio 150
4 Nombre Camila Prado - Codigo de asiento B-39 - Precio 250
5 Nombre Luis Choque - Codigo de asiento B-45 - Precio 250
6 Nombre Ivan Condori - Codigo de asiento A-25 - Precio 500
7 Nombre Andrea Juan - Codigo de asiento C-120 - Precio 150
8 Nombre Jhoselin Lopez - Codigo de asiento C-110 - Precio 150
9 Nombre PAMELA - Codigo de asiento A-150 - Precio 500
10 Nombre RUBEN - Codigo de asiento B-57 - Precio 250
11 Nombre MARCELO - Codigo de asiento C-114 - Precio 150
```



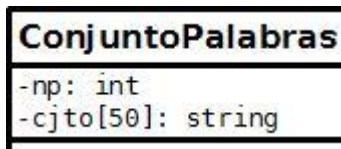
```
<terminated> Principal (10) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (17/09/2016 17:56:25)
Nombre del teatro 16 de Julio numero de entradas vendidas 8
1 Nombre Pedro Perez - Codigo de asiento A-20 - Precio 500
2 Nombre Juana Alarcon - Codigo de asiento C-85 - Precio 150
3 Nombre Jaime Quiroz - Codigo de asiento C-100 - Precio 150
4 Nombre Camila Prado - Codigo de asiento B-39 - Precio 250
5 Nombre Luis Choque - Codigo de asiento B-45 - Precio 250
6 Nombre Ivan Condori - Codigo de asiento A-25 - Precio 500
7 Nombre Andrea Juan - Codigo de asiento C-120 - Precio 150
8 Nombre Jhoselin Lopez - Codigo de asiento C-110 - Precio 150
**Buscar asientos caros vendidos
Nombre: Pedro Perez - Codigo de asiento {1}A-20
Nombre: Ivan Condori - Codigo de asiento {1}A-25
**Buscar persona x
La perona {0} si se encuentra en el teatro sus datos son:Pedro Perez
Codigo de asiento A-20 - Precio {1}500
**Vender asiento para una persona
Introduzca nombre de la persona su asiento y el precio de este.
PAMELA CHOQUE
A-15
500
**Vender asientos para n personas
¿A cuantas personas desea vender entradas?
2
Introduzca nombre de la persona su asiento y el precio de este.
RUBEN JARANDILLA
C-98
150
Introduzca nombre de la persona su asiento y el precio de este.
JOSE PACHECO
C-99
150
Nombre del teatro 16 de Julio numero de entradas vendidas 11
1 Nombre Pedro Perez - Codigo de asiento A-20 - Precio 500
2 Nombre Juana Alarcon - Codigo de asiento C-85 - Precio 150
3 Nombre Jaime Quiroz - Codigo de asiento C-100 - Precio 150
4 Nombre Camila Prado - Codigo de asiento B-39 - Precio 250
5 Nombre Luis Choque - Codigo de asiento B-45 - Precio 250
6 Nombre Ivan Condori - Codigo de asiento A-25 - Precio 500
7 Nombre Andrea Juan - Codigo de asiento C-120 - Precio 150
8 Nombre Jhoselin Lopez - Codigo de asiento C-110 - Precio 150
9 Nombre PAMELA CHOQUE - Codigo de asiento A-15 - Precio 500
10 Nombre RUBEN JARANDILLA - Codigo de asiento C-98 - Precio 150
11 Nombre JOSE PACHECO - Codigo de asiento C-99 - Precio 150
```

4. EJEMPLO SOBRECARGA DE OPERADORES CSharp C#

Dada la clase Conjunto de Palabras con atributos: *np* el número de palabras, *cjto[50]* el vector de cadenas, realizar las siguientes operaciones:

- Sobrecargar los operadores ++ y -- para leer y mostrar respectivamente un objeto Conjunto de Palabras
- Sobrecargar el método *Desplegar()*, para mostrar las cadenas cuya longitud sea mayor que n y mostrar las palabras palíndromes de un objeto Conjunto de Palabras.
- Sobrecargar el operador ~ para verificar si existen al menos 2 palabras repetidas.
- Sobrecargar el operador ^ para verificar si existe el elemento Z en 2 instancias del objeto Conjunto de Palabras.

Diagrama de clases



SOLUCION.-

ConjuntoPalabras.cs

```
using System;
```

```

public class ConjuntoPalabras
{
    private int np;
    private string [] cjto = new string[50];

    public int Np {
        get { return np; }
        set { np = value; }
    }
    public ConjuntoPalabras(){
        np = 0;
    }

    public string GetPalabra(int i){
        return cjto[i];
    }

    public void SetPalabra(int i, string x){
        cjto[i] = x;
    }
    //a
    public static ConjuntoPalabras operator ++(ConjuntoPalabras x){
        Console.WriteLine("Leyendo datos para el Conjunto de palabras...");
        Console.Write("Ingresar Nro. de palabras :");
        x.Np = int.Parse(Console.ReadLine());

        Console.WriteLine("Ingresar palabras : ");
        for (int i = 0; i < x.Np; i++)
            x.SetPalabra(i, Console.ReadLine());
        return x;
    }

    public static ConjuntoPalabras operator --(ConjuntoPalabras x){
        Console.WriteLine("Mostrando los datos del Conjunto de palabras...");
        Console.WriteLine("Nro. de palabras = "+x.Np);
        for (int i = 0; i < x.Np; i++)
            Console.WriteLine(x.GetPalabra(i));
        return x;
    }
    //b
    public void Desplegar(int n){
        Console.WriteLine("Palabras con longitud Mayor a "+n+": ");
        for(int i = 0; i < np; i++){
            if(cjto[i].Length > n){
                Console.WriteLine(cjto[i]);
            }
        }
    }

    public bool EsPalindrome(int i){
        int n = cjto[i].Length;
        for(int k = 0; k < n/2; k++){
            if(cjto[i][k] != cjto[i][n-i-1])
                return false;
        }
        return true;
    }

    public void Desplegar(){
        Console.WriteLine("Palabras palindromes en el Conjunto de palabras:");
        for(int i = 0; i < np; i++){
            if(EsPalindrome(i))
                Console.WriteLine(cjto[i]);
        }
    }
    //c
    public static ConjuntoPalabras operator ~(ConjuntoPalabras x){
        bool sw = true;
        for(int i = 0; i < x.Np; i++){
            for(int j = i+1; j < x.Np; j++){
                if(x.GetPalabra(i).Equals(x.GetPalabra(j))) {
                    Console.WriteLine("Existe palabras repetidas");
                }
            }
        }
    }
}

```

```

        return x;
    }
}
}
Console.WriteLine("No existen palabras repetidas");
return x;
}
//d
public bool BuscarCadena(String z){
    for(int i = 0; i < x.Np; i++){
        if(x.GetPalabra(i).Equals(Z))
            return true;
    }
    return false;
}

public static ConjuntoPalabras operator ^(ConjuntoPalabras x, ConjuntoPalabras y){
    Console.Write("Introduzca la cadena Z: ");
    string Z = Console.ReadLine();

    bool sw1 = x.BuscarCadena(Z);
    bool sw2 = y.BuscarCadena(Z);

    if(sw1 && sw2) Console.WriteLine("\nExiste la cadena "+Z+" en ambos objetos");
    else Console.WriteLine("No Existe la cadena "+Z+" en ambos objetos");

    return x;
}
}

```

Program.cs

```

using System;

class Program{
    public static void Main(string[] args){
        ConjuntoPalabras A = new ConjuntoPalabras();

        Console.WriteLine("\n                INCISO A");
        A++;
        A--;

        //Inciso b
        Console.WriteLine("\n                INCISO B");
        A.Desplegar(5);
        A.Desplegar();

        //Inciso c
        Console.WriteLine("\n                INCISO C");
        A = ~A;

        //Inciso d
        Console.WriteLine("\n                INCISO D");
        ConjuntoPalabras B = new ConjuntoPalabras();
        ConjuntoPalabras C = new ConjuntoPalabras();

        B++;

        C = A^B;

        Console.ReadKey(true);
    }
}

```

LA CAPTURA DE PANTALLA DE EJECUCION EN C#

```

C:\Users\Marcelo\Documents\SharpDevelop Proje...
Undo last action
INCISO A
Leyendo datos para el Conjunto de palabras...
Ingresar Nro. de palabras :4
Ingresar palabras :
Bolívar
Wilsterman
Aurora
Real Potosi
Mostrando los datos del Conjunto de palabras...
Nro. de palabras = 4
Bolívar
Wilsterman
Aurora
Real Potosi

INCISO B
Palabras con longitud Mayor a 5:
Bolívar
Wilsterman
Aurora
Real Potosi
Palabras palindromes en el Conjunto de palabras:

INCISO C
No existen palabras repetidas

INCISO D
Leyendo datos para el Conjunto de palabras...
Ingresar Nro. de palabras :4
Ingresar palabras :
San Jose
Strongest
Wilsterman
Guabira
Introduzca la cadena Z: Wilsterman
Existe la cadena Wilsterman en ambos objetos

```

PARTE II Ejercicio a resolver

Dadas las clases Torneo de deportes y Participante, donde los únicos deportes son [basquet,football, voley] , la matriz esta dada por el nombre del jugador y su especialidad, realizar los siguientes incisos con la sobrecarga de operadores y metodos:

- Instanciar un objeto de tipo torneo y otros dos del tipo participante.
- Implementar el operador "+" para aumentar los dos participantes al torneo.
- Implementar el operador "%" para saber cuántos participantes tiene cada deporte.
- Sobrecargar el metodo mostrar para mostrar la clase torneo y otro para mostrar a los jugadores del deporte x.

Torneo
- Nombre : string
- nroP : int
- jugador [100,2] : string
+ Torneo();
+ Operator +;
+ Operator %;
+ Mostrar();
+ Mostrar(string dep);

Participante
- Nombre : string
- Especialidad: string
+ Participante(string dep);
+ getDep();
+ getNom();

Participante.cs

```

public class Participante{
    private string nombre;
    private string deporte;

    public Participante(string nom,string dep){
        this.nombre = nom;
        this.deporte = dep;
    }
    public string Nombre {
        get { return nombre; }
    }
    public string Deporte {
        get { return deporte; }
    }
}

```

Torneo.cs

```

public class Torneo{
    private string nombre;
    private int nroP;
    private string [,] jugador = new string[100,3];

    public Torneo(string nom){
        this.nombre = nom; this.nroP = 5;
        this.jugador[1,1]="Roberto";this.jugador[1,2]="Footbol";
        this.jugador[2,1]="Camila";this.jugador[2,2]="Basquet";
        this.jugador[3,1]="carla";this.jugador[3,2]="Voley";
        this.jugador[4,1]="Manuel";this.jugador[4,2]="Footbol";
        this.jugador[5,1]="Fernando";this.jugador[5,2]="Voley";
    }
    public void Mostrar(){
        Console.WriteLine("\n\nNOMBRE DEL TORNEO: {0} NUMERO DE PARTICIPANTES {1}\n",this.nombre,this.nroP);
        for(int i=1 ; i<= this.nroP ; i++){
            Console.WriteLine(i+" MONBRE {0} DEPORTE {1}",this.jugador[i,1],this.jugador[i,2]);
        }
    }
    public static Torneo operator % (Torneo t, Participante p1){
        t.nroP++;
        t.jugador[t.nroP,1]=p1.Nombre;
        t.jugador[t.nroP,2]=p1.Deporte;
        return t;
    }
    public static Torneo operator ++ (Torneo t){
        string [] vd = new string[4];
        int [] tt = new int[4];
        vd[1]="Footbol";vd[2]="Basquet";vd[3]="Voley";
        tt[1]=0;tt[2]=0;tt[3]=0;
        for(int i=1 ; i<= t.nroP ; i++){
            string d = t.jugador[i,2];
            for(int j=1 ; j<=3 ; j++){
                if(d == vd[j])
                    tt[j]=tt[j]+1;
            }
        }
        for(int j=1 ; j<=3 ; j++){
            Console.WriteLine("\n"+vd[j]+" = "+tt[j]);
        }
        return t;
    }
    public void Mostrar(string d){
        Console.WriteLine("\n\nJUGADORES DE :"+d+"");
        for(int i=1 ; i<= this.nroP ; i++){
            string a = this.jugador[i,2];
            if(d == a)
                Console.WriteLine("\nJUGADOR: "+this.jugador[i,1]);
        }
    }
}

```

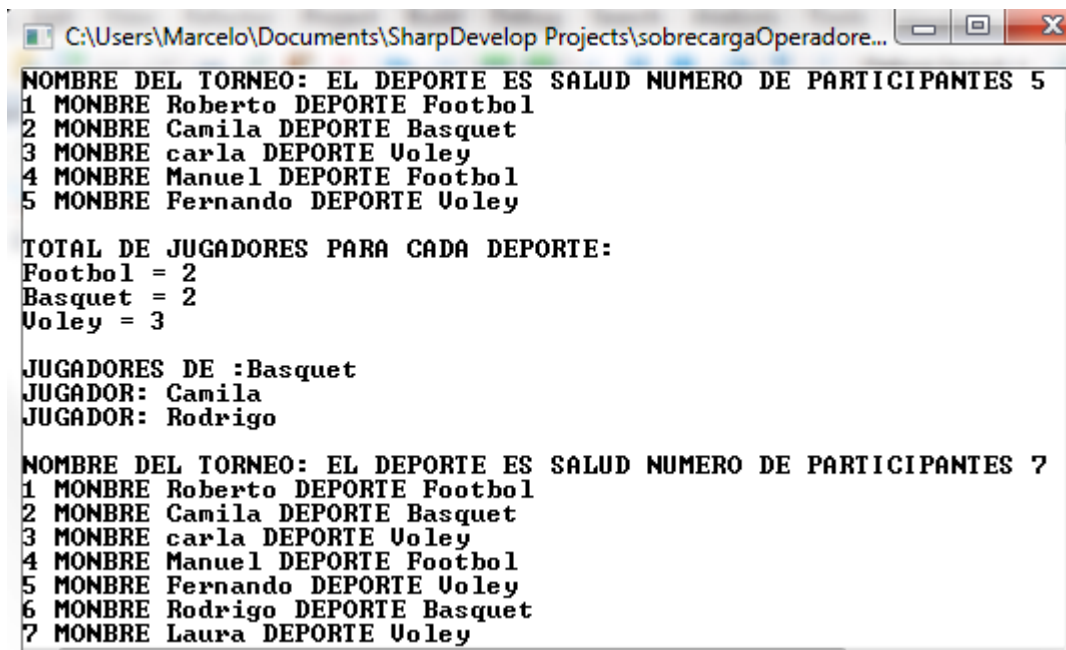
Program.cs

```

public class Program{
    public static void Main() {
        Torneo t = new Torneo("EL DEPORTE ES SALUD") ;
        Participante p1 = new Participante("Rodrigo", "Basquet");
        Participante p2 = new Participante("Laura", "Voley");
        t.Mostrar();
        t=t%p1;
        t=t%p2;
        Console.Write("\nTOTAL DE JUGADORES PARA CADA DEPORTE: ");
        t++;
        t.Mostrar("Basquet");
        t.Mostrar();
        Console.ReadKey();
    }
}

```

Captura de Pantalla



EJERCICIOS.

- Sobrecargar el operador "!", para ordenar Ordenar el Torneo por orden alfabético
- Instanciar otro Torneo con 4 participantes, sobrecargar el operador ">", para determinar cual Torneo es el que tiene más participantes en el deporte Z.
- Sobrecargar el operador "<=", para que dado un Torneo, genere un vector de participantes del deporte Z.