

Nombre: Andrés Alejandro Cárdenas de la Cruz		Matricula: 2896424
Nombre del curso: Computación En Java	Nombre del profesor: Manuel Cruz Serrano	
Tema 13. Expresiones Lambda	Actividad 13. Clases anónimas y expresiones lambda	
Fecha: 28/11/2020		

Solución Implementada

```
package sort;

import java.util.List;

public class Principal {

    public static void main(String[] args) {
        //instancias de clase
        Metodos metodos = new Metodos();
        //lista creada a partir de un metodo
        List<String> list = metodos.agregarDatos();
        //instancias de clase
        ClaseAnonima claseAnonima = new ClaseAnonima(list);
        Lamda lamda = new Lamda(list);
        MetodoReferencia metodoReferencia = new MetodoReferencia(list);
        //imprime las lista a partir de clases en las instancias
        System.out.println("Lista actual\n"+list);
        System.out.println("Lista Ordenada Alfabeticamente Por Clases Anonimas
\n" + claseAnonima.Alfabeticamente() );
        System.out.println("Lista Ordenada Por Longitud Por Clases Anonimas \n"
+ claseAnonima.longitud()+ "\n\n");
        System.out.println("Lista actual\n"+list);
        System.out.println("Lista Ordenada Alfabeticamente Por Lamda \n" +
lamda.Alfabeticamente() );
        System.out.println("Lista Ordenada Por Longitud Por Lamda \n" +
lamda.longitud()+ "\n\n");
        System.out.println("Lista actual\n"+list);
        System.out.println("Lista Ordenada Alfabeticamente Por Metodos De
Referencia \n" + metodoReferencia.Alfabeticamente() );
        System.out.println("Lista Ordenada Por Longitud Por Metodos De
Referencia \n" + metodoReferencia.longitud());
    }
}
```

```
package sort;

import java.util.ArrayList;
import java.util.List;

public class Metodos {
    //retorna una lista con 10 datos
    public List<String> agregarDatos() {
        List<String> list = new ArrayList<>();
        list.add("zorro");
        list.add("lobo");
        list.add("pastor");
        list.add("gato");
        list.add("diego");
        list.add("roberto");
        list.add("santiago");
        list.add("oso");
        list.add("mariposa");
        list.add("perro");
        return list;
    }
}
```

```
package sort;

import java.util.Comparator;
import java.util.List;

public class ClaseAnonima {

    private List<String> lista;

    //constructor que inicializa una lista
    public ClaseAnonima(List<String> lista) {
        this.lista = lista;
    }

    //metodo de clase anonima que ordena alfabeticamente
    public List<String> Alfabeticamente(){
        Comparator<String> comparadorAlfa = new Comparator<String>() {
            @Override
            public int compare(String v1, String v2) {
                return v1.compareTo(v2);
            }
        };
        lista.sort(comparadorAlfa);
        return lista;
    }

    //metodo de clase anonima que ordena por longitud
    public List<String> longitud(){
        Comparator<String> comparadorLongitud = new Comparator<String>() {
            @Override
            public int compare(String v1, String v2) {
                return Integer.compare(v1.length(), v2.length());
            }
        };
        lista.sort(comparadorLongitud);
        return lista;
    }
}
```

```
package sort;

import java.util.Comparator;
import java.util.List;

public class MetodoReferencia {
    private List<String> lista;

    //constructor que inicializa una lista
    public MetodoReferencia(List<String> lista) {
        this.lista = lista;
    }

    //metodo de metodo de referencia que ordena alfabeticamente
```

```
        public List<String> Alfabeticamente() {
            Comparator<String> comparatorAlfa = String::compareTo;
            lista.sort(comparatorAlfa);
            return lista;
        }

        //metodo de metodo de referencia que ordena por longitud
        public List<String> longitud() {
            Comparator<String> comparatorLongitud =
            Comparator.comparingInt(String::length);
            lista.sort(comparatorLongitud);
            return lista;
        }
    }
}
```

```
package sort;

import java.util.List;

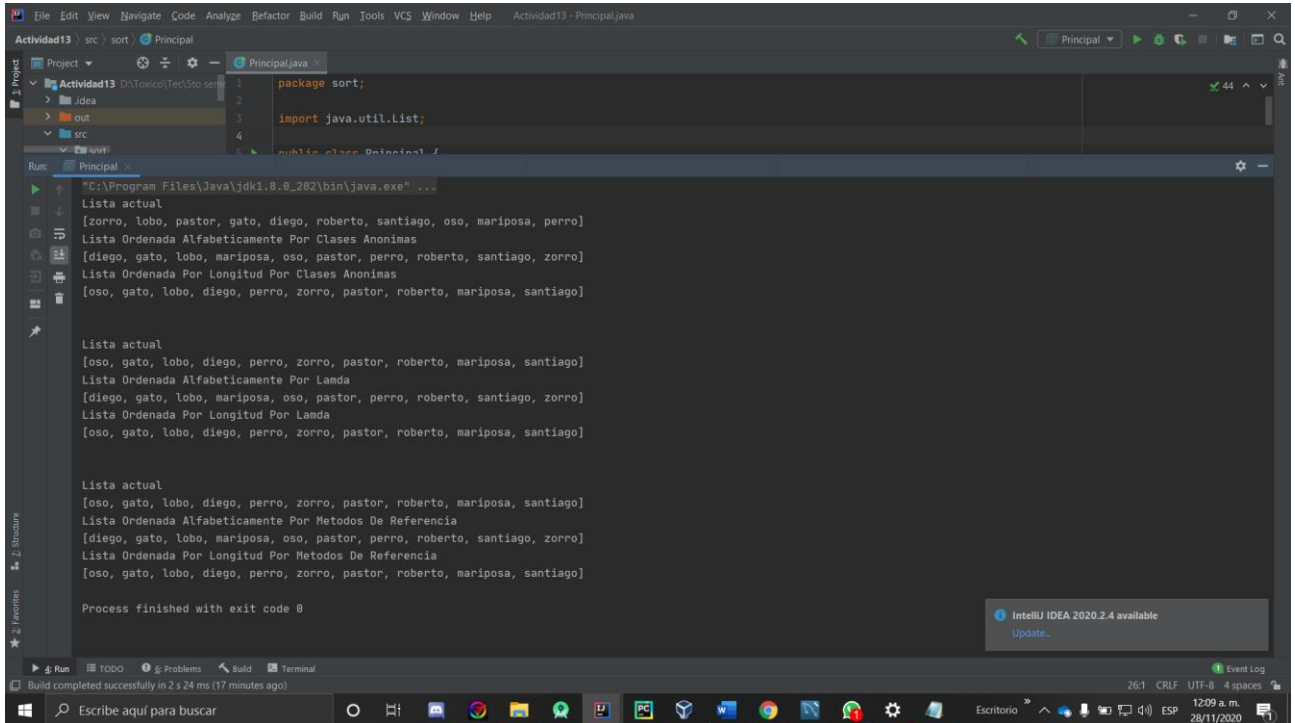
public class Lamda {

    private List<String> lista;

    //constructor que inicializa una lista
    public Lamda(List<String> lista) {
        this.lista = lista;
    }

    //metodo de Lamda que ordena alfabeticamente
    public List<String> Alfabeticamente() {
        lista.sort((v1, v2) -> v1.compareTo(v2));
        return lista;
    }

    //metodo de Lamda que ordena por longitud
    public List<String> longitud() {
        lista.sort((v1, v2) -> Integer.compare(v1.length(), v2.length()));
        return lista;
    }
}
```



```
package sort;

import java.util.List;

public class Principal {

    Lista actual
    [zorro, lobo, pastor, gato, diego, roberto, santiago, oso, mariposa, perro]
    Lista Ordenada Alfabeticamente Por Clases Anonimas
    [diego, gato, lobo, mariposa, oso, pastor, perro, roberto, santiago, zorro]
    Lista Ordenada Por Longitud Por Clases Anonimas
    [oso, gato, lobo, diego, perro, zorro, pastor, roberto, mariposa, santiago]

    Lista actual
    [oso, gato, lobo, diego, perro, zorro, pastor, roberto, mariposa, santiago]
    Lista Ordenada Alfabeticamente Por Lambda
    [diego, gato, lobo, mariposa, oso, pastor, perro, roberto, santiago, zorro]
    Lista Ordenada Por Longitud Por Lambda
    [oso, gato, lobo, diego, perro, zorro, pastor, roberto, mariposa, santiago]

    Lista actual
    [oso, gato, lobo, diego, perro, zorro, pastor, roberto, mariposa, santiago]
    Lista Ordenada Alfabeticamente Por Metodos De Referencia
    [diego, gato, lobo, mariposa, oso, pastor, perro, roberto, santiago, zorro]
    Lista Ordenada Por Longitud Por Metodos De Referencia
    [oso, gato, lobo, diego, perro, zorro, pastor, roberto, mariposa, santiago]

    Process finished with exit code 0
```

Se implemento una solución apoyándome con 2 librerías siendo esta la librería `utils` y `comparator` con ayuda de los métodos `comparingInt` y `compareTo` apoyándome para realizar la comparación de cadenas en una haciendo un acomodo por orden alfabético y la otra por longitud retornando con enteros la longitud de estos

<https://github.com/alex-crdnz/actividades.git>