

Instituto Tecnológico de Tijuana

Nombre de Facultad

Ingeniería Informática



Proyecto / Tarea / Practica:

Práctica 3

Materia:

Datos Masivos

Facilitador:

Jose Christian

Alumnos:

Erik Saul Rivera Reyes

Brayan Baltazar Moreno

Fecha:

Tijuana Baja California a 08 de 03 2022

Code

```
//Practica 3

C:\Spark\spark-3.2.1-bin-hadoop3.2\bin\spark-shell

//Parte 1
val Lista=List ( "rojo", "blanco", "negro")

//Parte 2
val Lista = List("rojo","blanco","negro")
val Lista2 = "verde,amarillo,azul,naranja,perla" :: Lista

//Parte 3
import scala.collection.mutable.ListBuffer

var Lista = new ListBuffer[String]()
Lista += "rojo"
Lista += "blanco"
Lista += "negro"
Lista += "verde"
Lista += "amarillo"
Lista += "azul"
Lista += "naranja"
Lista += "perla"
println(Lista(3) + " " + Lista(4) + " " + Lista(5))

//Parte 4
var a = 1;
do
{
    print(a + " ");
    a = a + 5;
}while(a <= 1000);

//Parte 5
val x = List(1,3,3,4,6,7,3,7)
x.distinct

//Parte 6
val mapMut = scala.collection.mutable.Map("Jose" -> 20,
                                           "Luis" -> 24,
                                           "Susana" -> 27)

println("Mapa: " + mapMut)
mapMut("Miguel") = 23
println("Agregando a Miguel " + mapMut)
```

1-Se creo una lista con los valores tipo string pedidos para después desplegarla

```
scala> val Lista=List ( "rojo", "blanco", "negro")
Lista: List[String] = List(rojo, blanco, negro)

scala> _
```

2-Se crean dos listas las cuales se concatenan para así generar una lista con todos los valores de ambas

```
scala> val Lista = List("rojo","blanco","negro")
Lista: List[String] = List(rojo, blanco, negro)

scala> val Lista2 = "verde,amarillo,azul,naranja,perla" :: Lista
Lista2: List[String] = List(verde,amarillo,azul,naranja,perla, rojo, blanco, negro)

scala> _
```

3-Se crea una lista mutable para así desplegar mi lista con los valores pedidos a diferencia de las listas anteriores esta me permite modificarla y desplegar solamente los valores tipo string que necesito

```
scala> var Lista = new ListBuffer[String]()
Lista: scala.collection.mutable.ListBuffer[String] = ListBuffer()

scala> Lista += "rojo"
res0: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo)

scala> Lista += "blanco"
res1: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco)

scala> Lista += "negro"
res2: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro)

scala> Lista += "verde"
res3: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro, verde)

scala> Lista += "amarillo"
res4: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro, verde, amarillo)

scala> Lista += "azul"
res5: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro, verde, amarillo, azul)

scala> Lista += "naranja"
res6: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro, verde, amarillo, azul, naranja)

scala> Lista += "perla"
res7: scala.collection.mutable.ListBuffer[String] = ListBuffer(rojo, blanco, negro, verde, amarillo, azul, naranja, perla)

scala> println(Lista(3) + " " + Lista(4) + " " + Lista(5))_
verde amarillo azul

scala>
```

4-Se utilizo un ciclo repetitivo do/while el cual tenia como condicion que mientras los valores fueran iguales o menores a 1000 la operación de sumar de 5 en 5 a nuestra variable x se repetira

```
scala> do
  {
    print(a + " ");
    a = a + 5;
  } while(a <= 1000);
1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96 101 106 111 116 121 126 131 136 141 146 151 156 161 166 171 176 181 186 191 196 201 206 211 216 221 226 231 236 241 246 251 256 261 266 271 276 281 286 291 296 301 306 311 316 321
326 331 336 341 346 351 356 361 366 371 376 381 386 391 396 401 406 411 416 421 426 431 436 441 446 451 456 461 466 471 476 481 486 491 496 501 506 511 516 521 526 531 536 541 546 551 556 561 566 571 576 581 586 591 596 601 606 611 616
621 626 631 636 641 646 651 656 661 666 671 676 681 686 691 696 701 706 711 716 721 726 731 736 741 746 751 756 761 766 771 776 781 786 791 796 801 806 811 816 821 826 831 836 841 846 851 856 861 866 871 876 881 886 891 896 901 906 911 9
96 921 926 931 936 941 946 951 956 961 966 971 976 981 986 991 996
scala>
```

5- Se crea una lista con valores numericos especificados para despues utilizar la herramienta "distinct" para asi solamente desplegar los valores unicos sin ningun duplicado

```
scala> val x = List(1,3,3,4,6,7,3,7)
x: List[Int] = List(1, 3, 3, 4, 6, 7, 3, 7)

scala> x.distinct
res9: List[Int] = List(1, 3, 4, 6, 7)

scala>
```

6-Se crea un mapa mutable al cual se le introducen varios nombres con edades iniciales para despues actualizarlo con un nuevo nombre y desplegar tanto el mapa inicial como el actualizado con una nueva persona

```
scala> val mapMut = scala.collection.mutable.Map("Jose" -> 20,
  | "Luis" -> 24,
  | "Susana" -> 27)
mapMut: scala.collection.mutable.Map[String,Int] = Map(Susana -> 27, Luis -> 24, Jose -> 20)

scala> println("Mapa: " + mapMut)
Mapa: Map(Susana -> 27, Luis -> 24, Jose -> 20)

scala> mapMut("Miguel") = 23

scala> println("Agregando a Miguel " + mapMut)
Agregando a Miguel Map(Susana -> 27, Miguel -> 23, Luis -> 24, Jose -> 20)

scala>
```