# Ejercicios y Repaso

## Problemas

Elizabeth León Guzmán, Ph.D.
eleonguz@unal.edu.co

Jonatan Gómez Perdomo, Ph. D.
jgomezpe@unal.edu.co

Arles Rodríguez, Ph.D.
aerodriguezp@unal.edu.co

Camilo Cubides, Ph.D. (c)
eccubidesg@unal.edu.co

Carlos Andres Sierra, M.Sc.
casierrav@unal.edu.co

Research Group on Data Mining – Grupo de Investigación en Minería de Datos – (Midas)
Research Group on Artificial Life – Grupo de Investigación en Vida Artificial – (Alife)
Computer and System Department
Engineering School
Universidad Nacional de Colombia

# Agenda




MIDAS

UNIVERSIDAD NACIONAL DE COLOMBIA un

# Problema I

## Problema

Se quiere realizar un programa orientado por objetos que permita optimizar una función asociada a un tipo de dato particular (el cual puede ser un real, entero, arreglo de binarios o cadena de caracteres) y que devuelve un real, usando el método del ascenso a la colina.
Código disponible en el repositorio:
https://github.com/jgomezpe/ascensocolina

# Agenda




MIDAS

UNIVERSIDAD NACIONAL DE COLOMBIA un

# Problemas varios

## Problemas

1. Dada una lista de números, eliminar los duplicados e imprimir la lista procesada.
2. Dadas dos listas de números, las cuales pueden ser de distinto tamaño, e imprimir los números se encuentran en ambas listas.
3. Dada una lista de objetos, imprimir la inversa de dicha lista.
4. Desarrollar un algoritmo que verifique si todas las parejas `clave:valor` de un HashMap se encuentran en otro HashMap.
5. Desarrollar una función que reciba dos TreeMap como parámetros y los mezcle, es decir, que se construya un nuevo TreeMap con las llaves de los dos TreeMap recibidos; si hay una `clave` repetida en ambos TreeMap, se debe asignar el `valor` que tenga la `clave` en el primer TreeMap.

# Agenda



# Ejercicio (I)

## Problemas

Implemente la Clase `Calculadora`, que tendrá un método por cada operación matemática básica (suma, resta, multiplicación, división, módulo).
Luego cree la Clase `CalculadoraTest`, e implemente un método de Testing por cada método de la clase `Calculadora`.
Prepare casos de prueba para cada operación:

- ¿Qué pasa cuando la división es sobre cero?
- ¿Qué pasa en el módulo con cero?
- ¿Qué ocurre cuando los números son muy grandes?
- ¿Qué ocurre cuando los números son muy pequeños?

# Ejercicio (II)

## Problemas (continuación)

Implemente la clase `Persona`. Tendrá como atributos nombres, apellidos, nombreCompleto, edad, peso y estatura. El constructor recibirá nombres, apellidos, edad, peso y estatura y calculará el nombreCompleto como la concatenación de los nombres con los apellidos. Tendrá un único método `calcularIMC()` que retornará el IMC de la persona.
Cree la clase `PersonaTest` e implemente dos métodos de Testing, uno para evaluar el nombreCompleto y otro para evaluar el IMC. Defina los casos de prueba y corra los tests.
¿Pasaron todos los test a la primera? ¿Qué salió mal?