

# TICS311: Tarea #4

## Universidad Adolfo Ibañez

Ahmad Armoush  
ahmad.armoush@edu.uai.cl

Danilo Bórquez Paredes  
danilo.borquez.p@uai.cl

Wilson González  
wilson.gonzalez@edu.uai.cl

Miguel Romero  
miguel.romero.o@uai.cl

7 de junio de 2022

## Objetivos

- Internalizar conceptos de árboles
- Uso de archivos de cabecera

### 1. La recta final

Ya estamos acabando con el enemigo. Nuestros espías recopilan cada vez mejor información, por lo que ahora el archivo que le entregan no tiene campos vacíos, como se muestra a continuación.

```
mensaje.txt

first_name,danger_category,attack_prob
Kearney Carding,1,0.2733129
Antonin Revans,1,0.123584
Conney Lochrie,5,0.694532
Anton Sellack,2,0.0023549
Lory Brydie,3,0.3140912
```

Usted debe guardar la información de este archivo en un **árbol binario de búsqueda AVL**, ordenado según categoría, y luego según probabilidad de ataque.

El programa debe leer el archivo con los datos de entrada, guardarlos en la estructura definida y mostrar por pantalla los nombres de las  $N$  personas más peligrosas.

**No puede usar arreglos ni listas, exceptuando un arreglo de caracteres para leer el archivo.**

#### 1.1. Objetivo

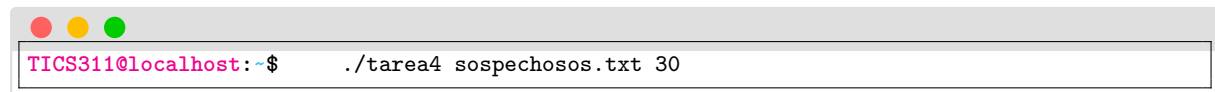
Generar la estructura de datos mencionada arriba, y mostrar por pantalla el nombre de las  $N$  personas más peligrosas. Como refresco de memoria, se adjuntan los criterios:

- Un número mayor de danger\_category implica mayor peligrosidad
- Un número mayor de attack\_prob implica mayor peligrosidad
- Si existen dos nombres que empatan en peligrosidad (danger\_category y attack\_prob), entonces se ordenarán por orden alfabético.
- Primero se ordena por categoría, luego por probabilidad

## 1.2. Entrada del programa

Su programa recibirá como entrada sólo dos parámetros, correspondientes al nombre del archivo que contiene la lista de nombres, y la cantidad de nombres que se quiere mostrar

A continuación un ejemplo de ejecución:

A terminal window with a title bar containing three colored circles (red, yellow, green). The terminal text shows a user at a TICS311@localhost prompt running the command './tarea4 sospechosos.txt 30'.

El ejemplo mostrado más arriba utilizará el archivo *sospechosos.txt* que contiene la lista de nombres, y generará la estructura de datos mencionada en el inicio con 30 de esos nombres. El archivo de entrada siempre tendrá **a lo menos** la cantidad de nombres que se piden.

## 1.3. Salida del programa

El programa debe mostrar por consola los nombres (y sólo los nombres!) más peligrosos de la lista entregada

## 2. Sobre la entrega

- La tarea debe ser hecha en lenguaje de Programación C.
- Cada grupo puede ser de 2 o 3 personas.
- Debe trabajar la tarea con árboles AVL.
- **Se evaluará orden y modularidad de su código.**
- La tarea se debe entregar el día **Domingo 19 de Junio** a las 23:59.
- Por cada día de atraso se descontará 1 punto, comenzando a las 00:00 horas del siguiente día. Por ejemplo si entrega la tarea a las 00:00 del siguiente día, la nota máxima que puede obtener es un 6.0
- Para la corrección se utilizará un compilador gcc v 5.1 o superior
- La entrega se realiza por la plataforma Webcursos<sup>1</sup>
- El archivo a entregar debe ser un zip que contenga una carpeta en su interior (y sólo una carpeta) con el nombre **tarea4**. Dentro de esa carpeta debe haber un Makefile<sup>2</sup> y por lo menos un main.c. Además el nombre de su zip debe ser grupoX-tarea4.zip, donde X es el número de su grupo.
- **IMPORTANTE:** el directorio **tarea4** debe contener el Makefile que generará el archivo ejecutable **tarea4**.

---

<sup>1</sup><http://webc.uai.cl>

<sup>2</sup>Este archivo deben generarlo. Información útil pueden encontrarla en <https://stackoverflow.com/questions/1484817/how-do-i-make-a-simple-makefile-for-gcc-on-linux>. Pueden probar su Makefile en un computador con MAC o Linux, o en VSCode en la terminal con UBUNTU. También puede probarse en la shell de <https://repl.it/>