



Práctica 10: Estructuras de tipo Registro

1. Escriba las expresiones para los siguientes items:
 - a. Defina el tipo `dino` como la estructura con los siguientes campos:
 - `Nombre` (`string`)
 - `Peso` (`real`, peso en toneladas)
 - `Altura` (`real`, altura en pies)
 - b. Cree una estructura llamada `tyranno`, de tipo `dino`, que represente el hecho que los `tyrannosaurus` pesaban 7 toneladas y medían 20 pies de alto
 - c. Cree una estructura llamada `bracchio`, de tipo `dino`, que represente el hecho que los `brachiosaurus` pesaban 50 toneladas y medían 40 pies de alto
 - d. Escriba una expresión para obtener la altura del `tyranno`
 - e. Escriba una expresión para obtener el peso del `bracchio`
2. Escriba las siguientes funciones basándose en el tipo de estructura `dino`:
 - a. Dada una lista de estructuras `dino`, encontrar el más alto de la lista
 - b. Dada una lista de estructuras `dino`, encontrar el más pesado de la lista
 - c. Dada una lista de estructuras `dino`, encontrar el más alto de la lista
3. Trabajaremos ahora con la estructura estudiante:

```
type estudiante = {ID: int, Cursos: string list, Nombre: String};
```

- a. Dada una lista de estudiantes y un nombre `n`, encontrar todas las estructuras con el valor `n` en el campo `nombre`
 - b. Dada una lista de estudiantes y un ID `i`, devuelva la lista de cursos del estudiantes correspondiente
 - c.
 - d. Dada una lista de estudiantes y un curso `c`, encontrar los nombres de los estudiantes que están inscriptos a dicho curso
4. Considere la siguiente definición de árbol binario ya vista en clase (seguimos trabajando con el tipo de dato `estudiante`):

```
- datatype 'etiqueta arbolbin =  
  Vacio |  
  Nodo of 'etiqueta * 'etiqueta arbolbin * 'etiqueta arbolbin;
```

Suponga que existe una agenda de estudiantes implementada como un árbol binario de búsqueda de tipo `estudiante arbolbin`. Cada nodo del árbol es una estructura que contiene los datos de un estudiante (el nombre del alumno escrito todo en minúscula), su ID y los cursos a los cuales asiste. El árbol binario de búsqueda está ordenado de acuerdo al orden alfabético del campo `nombre`. Implemente las siguientes funcionalidades que permitan utilizar la agenda:

- a. Defina una función `ingresar-contacto` que tome una agenda (implementada como un árbol binario de búsqueda) y los datos de una persona y si dicha persona no se encuentra ya en la agenda se agrega a la misma (recuerde mantener la propiedad de árbol binario de búsqueda). Si la persona ya se encuentra en la agenda no debe modificar la agenda.

Ayuda: Para esto defina antes una función que dados los datos de un alumno, cree la estructura alumno con dichos datos.

- b. Defina una función **modificar-contacto** que tome una agenda de estudiantes (implementada como un árbol binario de búsqueda) y los datos de un estudiante y si dicha persona se encuentra ya en la agenda modifica los datos asociados. Si el estudiante no se encuentra en la agenda no debe modificar la agenda.
 - c. Defina una función **buscar-cursos** que tome una agenda (implementada como un árbol binario de búsqueda) y el nombre de un alumno y devuelva los cursos a los cuales asiste dicho alumno. Si el alumno no se encuentra en la agenda debe devolver el mensaje “No se encontró el estudiante buscado”.
 - d. Defina una función **buscar-ID** que tome una agenda (implementada como un árbol binario de búsqueda) y el nombre de un estudiante y devuelva su ID en caso de que dicho alumno esté en la agenda. Si estudiante no se encuentra en la agenda debe devolver ~ 1 .
5. Para resolver los siguientes items tendrá que usar las funciones de alto orden que provee ML (**map**, **filter**, **foldr**, **foldl**). Seguimos trabajando con el tipo de dato **estudiante** y **arbolbin**.
- a. Una función **estudiantes_no_inscriptos** que dada una lista de estudiantes y un curso **c**, devuelva los nombres de aquellos estudiantes no inscriptos a dicho curso
 - b. Una función **cant_estudiantes_no_inscriptos** que dada una lista de estudiantes y un curso **c**, devuelva la cantidad de estudiantes no inscriptos a dicho curso
 - c. Una función **estudiantes_no_inscriptos_v2** que dado un **estudiante arbolbin** y un curso **c**, devuelva los nombres de aquellos estudiantes no inscriptos a dicho curso.
 - d. Una función **cant_estudiantes_no_inscriptos_v2** que dado un **estudiante arbolbin** y un curso **c**, devuelva la cantidad de estudiantes no inscriptos a dicho curso