Algoritmos y Estructuras de Datos I

Examen final del Taller para condición LIBRE

Funcional: Haskell

Vamos a representar un *tren de carga* usando *Haskell* y para ello se deben definir nuevos tipos:

- Primeramente se debe definir el tipo `Item` que tiene constructores `Azucar`, `Cafe`, `Maiz`, `Trigo` y `Yerba`, todos ellos sin parámetros. El tipo `Item` **no debe pertenecer** a la clase `Eq`.
- Luego se debe definir el tipo `Toneladas` como sinónimo del tipo `Int`.
- El tipo `Cargamento` debe tener dos constructores:
  - Constructor `SinCarga`: No tiene parámetros y representa el cargamento vacío.
  - Constructor `Carga`: Tiene dos parámetros, el primero de tipo `Item` (indica el tipo de carga que tiene el cargamento) y el segundo parámetro es de tipo `Toneladas` (el peso de ese cargamento).
- El tipo `Numeracion` que debe ser un sinónimo de `Int`
- El tipo `Tren` que tiene dos constructores:
  - Constructor `Vagon`: Tiene tres parámetros, el primero de tipo `Numeracion` (la numeración del vagón), el segundo de tipo `Cargamento` (el cargamento que lleva ese vagón) y el tercero de tipo `Tren` que es el resto del tren.
  - Constructor `Fin`: No tiene parámetros y representa el final del tren.

Asegurarse que los tipos `Tren`, `Cargamento` e `Item` estén en la clase `Show`

a) Programar la función:

```
vagones_item :: Tren -> Item -> [Numeracion]
```

que dado un tren `ts` y un ítem `i`, devuelve las numeraciones de los vagones del tren `ts` que transportan el ítem `i`.

b) Dar una expresión de tipo `Tren` que tenga al menos tres elementos donde se utilicen los dos constructores del tipo.

Imperativo: C

Hacer un programa que verifique si el elemento ubicado en el índice `k` de un arreglo es el máximo o si es el mínimo. Para ello programar la siguiente función

```
struct s_minmax_t verificar_minmax(int a[], int tam, int k);
```

donde la estructura `struct s_minmax_t` se define de la siguiente manera:

```
struct s_minmax_t {
    bool es_maximo;
    bool es_minimo;
};
```

La función toma un arreglo `a[]`, su tamaño `tam` y un índice `k` y debe devolver una estructura con dos *booleanos* que respectivamente indican:

- Todos los elementos de `a[]` son menores o iguales al elemento ubicado en la posición `k` (se guarda en `es_maximo`)
- Todos los elementos de `a[]` son mayores o iguales al elemento ubicado en la posición `k` (se guarda en `es_minimo`)

La función `verificar_minmax` debe implementarse con un único ciclo y **no debe mostrar mensajes** por pantalla **ni pedir valores al usuario**.

En la función `main` se debe solicitar al usuario ingresar un arreglo de longitud `N`, donde `N` debe definirse como una constante. **El usuario no debe elegir el tamaño del arreglo**. Luego se debe pedir el índice `k` y verificar con `assert` que `k` es un número mayor que `0` y menor que `N`. Finalmente mostrar el resultado de la función `verificar_minmax` por pantalla (los dos valores de la estructura).