Trabajo Práctico Integrador Nro. 2

Unidad 4: Paradigma Funcional

Objetivo:

El objetivo del presente trabajo práctico integrador es la evaluación de los temas de la unidad número 4: Paradigma de Programación Funcional, que se detallan a continuación: definición de funciones; uso de expresiones en Haskell tales como: guardas, if – then – else, case of, otherwise, where, entre otras; listas por comprensión; recursividad; tuplas.

Enunciado

Evaluación de Programación Funcional

Una heladería tiene a la venta diferentes presentaciones de sus productos de helados artesanales. A continuación, se muestra en la ***tabla 1: tipos de presentaciones de helados***, los siguientes datos: los códigos de cada presentación, su descripción y el precio de venta por unidad.

Tabla 1. Tipos de presentaciones de helados

Código	Descripción	Precio de venta unitario [$].
'S'	"cono simple"	2500.0
'D'	"cono doble"	3500.0
'T'	"cono triple"	4000.0
'P'	"pinito gigante"	5900.0

Su tarea:

Codificar en Haskell las siguientes funciones:

1) Implementar una función que reciba como parámetro el código de un tipo de presentación de helado y retorne el precio de venta unitario correspondiente, según los datos de la ***tabla 1: Tipos de presentaciones de helados***. En caso de recibirse un código de tipo de presentación de helado que no figure en la tabla, la función deberá devolver 0.0. Utilizar expresión case of. Nombre sugerido de la función: ***precioPorÇodigo***

Ejemplos:

```
Main> precioPorCodigo'S'
      2500.0
Main> precioPorCodigo'D'
      3500.0
Main> precioPorCodigo'T'
      4000.0
Main> precioPorCodigo'P'
      5900.0
Main> precioPorCodigo'X'
      0.0
```

2) Implementar una función que reciba dos parámetros: una lista de códigos de tipos de presentaciones de helados y un precio de referencia. La función deberá devolver la cantidad de códigos de tipos de presentaciones de helados que figuren en la lista recibida como primer parámetro, que tengan un precio de venta unitario **mayor** al precio de referencia recibido como segundo parámetro. Reutilizar la función del punto 1). Nombre sugerido de la función: ***cantidadDeHeladosMasCarosQue***

Ejemplos:

```
Main> cantidadDeHeladosMasCarosQue ['S','T','P','D','S'] 3800.0
      2
Main> cantidadDeHeladosMasCarosQue ['S','T','P','D','S'] 1800.0
      5
Main> cantidadDeHeladosMasCarosQue ['S','T','P','D','S'] 4500.0
      1
```

```
Main> cantidadDeHeladosMasCarosQue['S','T','P','D','S'] 6500.0
      0
Main> cantidadDeHeladosMasCarosQue['S','T','P','D','S'] 2750.0
      3
```

3) Implementar una función que reciba 3 parámetros: una lista de precios, un valor de referencia ***min***, y un valor de referencia ***max***.

La función deberá devolver una lista con los precios de la lista recibida como primer parámetro, tales que sean mayores o iguales al valor de referencia ***min*** y menores o iguales al valor de referencia ***max***. Plantear dos funciones que lo resuelvan, una de ellas deberá estar implementada utilizando recursividad.

Nombres sugeridos: preciosEntre y preciosEntreRecursivo

Ejemplos:

```
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 3200.0 4500.0
      [3500.0,4000.0]
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 1200.0 4500.0
      [2500.0,3500.0,4000.0]
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 3200.0 6500.0
      [3500.0,4000.0,5900.0]
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 1200.0 6500.0
      [2500.0,3500.0,4000.0,5900.0]
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 1200.0 1800.0
      []
Main> preciosEntre[2500.0,3500.0,4000.0,5900.0] 4200.0 5900.0
      [5900.0]
Main> preciosEntre[] 4200.0 5900.0
      []
```

4) Implementar una función que reciba como parámetro el código de un tipo de presentación de helado y retorne una tupla de dos elementos. De acuerdo con la ***tabla 1: tipos de presentaciones de helados***, el primer elemento de la tupla deberá ser la descripción del tipo de presentación correspondiente al código recibido como parámetro, y el segundo elemento de la tupla deberá ser el precio de venta unitario del tipo de presentación correspondiente al código recibido como parámetro. En caso de recibirse un código de tipo de presentación de helado que no figure en la tabla, la función deberá devolver la tupla ("",0.0).

Ejemplos:

```
Main> informacionPorCodigo 'S'
("cono simple",2500.0)
Main> informacionPorCodigo 'D'
("cono doble",3500.0)
Main> informacionPorCodigo 'T'
("cono triple",4000.0)
Main> informacionPorCodigo 'P'
("pinito gigante",5900.0)
Main> informacionPorCodigo 'X'
("",0.0)
```

Tabla de puntuación de los ítems evaluados

Nro. de ítem	Ítem o requerimiento a evaluar	Puntaje	Observaciones	Obtenido
1	Función nro. 1.	20		
2	Función nro. 2.	30		
3	Función nro. 3.	30		
4	Función nro. 4.	20		
	Total	**100**		