

PROGRAMACION DECLARATIVA

TRABAJO PRÁCTICO Nº1

**ALUMNO: Johana Testa**

# Introducción

El objetivo de este Trabajo Práctico es revisar los temas abordados durante este

Período sobre Programación Funcional en Haskell, implementando un pequeño

Software para procesar un archivo de texto

.

**1) Módulo para el dato tipo Hora**

Se debe implementar el tipo de dato Hora y las funcionalidades asociadas para gestionar

los registros de entrada/salida del personal como un módulo Haskell.

Implemente lo siguiente:

1. Cada dato tipo Hora se conformará con dos valores tipo Int,

2. Modifique la instancia Show para imprimir las horas y minutos (HH:MM),

3. El constructor del dato,

4. Los getters para hora y minuto,

5. Las funciones para normalización y desnormalización a minutos desde la hora cero

(00:00) para facilitar la aritmética,

6. La función para calcular diferencia de tiempo.

**2) Programa principal y funciones auxiliares**

Se debe implementar la función main y las funciones auxiliares que considere necesarias

para leer el archivo de entrada y obtener información sobre los registros. Una vez

procesado el archivo, se deberá mostrar por pantalla lo siguiente:

1. Por cada registro, imprimir día de la semana, fecha y horas trabajadas (ver ejemplo

más abajo),

2. Cantidad de empleados (como entero),

3. Cantidad de horas acumuladas en el mes por todos los empleados (como flotante),

4. Cantidad de horas/hombre en el mes por todos los empleado

# Desarrollo

Este programa fue desarrollado totalmente en español para su mejor comprensión, primero comencé por releer “making your own types” en haskell.

Lo segundo que fui realizando fue analiza que funciones auxiliares iba a utilizar para poder leer correctamente el archivo de texto que nos fue brindado por el profesor, en cuanto a la definición del módulo hora fue bastante simple y sencillo de implementar.

En lo que respecta la función principal se utilizaron las siguientes funciones de las cuales voy a explicar un poco su funcionamiento para su mayor comprensión:

filtrarRenglones **::** **String** **->** **Bool**

Esta primera función filtra todos los renglones que pertenezcan a un dia en particular, caso contrario no imprime nada.

filtrarCampo **::** **Int** **->** **String** **->** **String**

En esta función se quitan los campos sobrantes para que la lectura del archivo sea mejor

filtraEmpleados **::** **String** **->** **Bool**

Esta filtra todos los empleadis para poder hacer el calculo a futuro de la cantidad de empleados.

creaLista **::** [**String**] **->** [**String**] **->** **Float** **->** ([**String**], **Float** )

Aquí se crea la lista o mejor dicho la estructura de datos que se va a mostrar

devuelveFloat **::** **Int** **->** **Float**

esta función toma un entero y devuelve un float para poder devolver el promedio y la cantidad de horas hombre.

crearHoras **::** **Int** **->** **String** **->** **Hora**

Se crean las horas para poder trabajar mejor con ellas

filtrarfechas **::** [**String**] **->** **String** **->** **Bool**

filtra todo lo que no sea una fecha

tomaHorasandMinutos **::** [**Char**] **->** **Hora**

se toman las horas y los minutos

# Conclusión

La mayor parte del trabajo estaba procesado en cuanto a la lectura de archivos pero fue difícil poder llegar a la cantidad de funciones que se requerían para poder correr el programa de manera correcta.