Algoritmos y Estructuras de Datos II

Parcial 28-04: Tema D - Demoras en Aeropuerto

Ejercicio 1: Implementación de Tabla de Demoras

En el directorio del ejercicio se encuentran los siguientes archivos:

Archivo	***Descripción***
`main.c`	Contiene la función principal del programa
`flight.h`	Declaraciones relativas a la estructura de los datos de vuelos y de funciones de carga y escritura de datos.
`flight.c`	Implementaciones **incompletas** de las funciones
`array_helpers.h`	Declaraciones / prototipos de las funciones que manejan la tabla de escalas
`array_helpers.c`	Implementaciones **incompletas** de las funciones que manejan el arreglo

Abrir el archivo **`inputs/example7055.in`** para ver cómo se estructuran los datos.

Cada línea contiene los datos de llegada de dos tipos de vuelos: `last_mile` (tramos finales) y `layover` (escalas). Primero aparecen **dos grupos de tres columnas** (seis columnas en total), las tres columnas de cada grupo se corresponden a la *hora*, *demora*, y *número de pasajeros*. El primer grupo representa la llegada para el tipo `last_mile`, y el segundo grupo la llegada para el tipo `layover`. El último dato es el código de vuelo, que debe ser guardado como un único char, ignorando los demarcadores `#`.

El esquema es el siguiente:

<hora_llegada>	<demora>	<pasajeros>	<hora_llegada>	<demora>	<pasajeros>	<código>

Consideraciones:

- La primera terna de datos siempre tiene tipo “0”: `last_mile`
- La siguiente terna de datos siempre tiene tipo “1”: `layover`
- A lo largo del día, solo hay una llegada para cada tipo de vuelo por hora.
- La demora está representada en minutos.
- Las horas siempre estarán en el rango 1 a 24 en los archivos de entrada.

El ejercicio consiste en completar el procedimiento de carga de datos en los archivos **`array_helpers.c`** y **`flight.c`**. Recordar que el programa tiene que ser *robusto*, es decir, debe tener un comportamiento bien definido para los casos en que la entrada no tenga el formato esperado.

Una vez completada la lectura de datos se puede verificar si la carga funciona compilando,

```
$ gcc -Wall -Werror -Wextra -pedantic -std=c99 -c array_helpers.c flight.c main.c
$ gcc -Wall -Werror -Wextra -pedantic -std=c99 array_helpers.o flight.o main.o -o delays
```

y luego ejecutar

```
$ ./delays inputs/example7055.in
```

Ejercicio 2: Análisis de los datos

Completar la siguiente función, definida en **`array_helpers`**

```
unsigned int compansation_cost(DelayTable a, unsigned int hour);
```

Esta función debe retornar el costo total que la empresa deberá pagar por las demoras del dia:

- El costo se calcula solo hasta las 6 de la tarde.
- Se paga un costo por cada minuto de demora que supera el límite permitido para un vuelo particular. Ese costo se abona a cada pasajero de ese vuelo.
- El costo a pagar por minuto está registrado como una constante: `COMPENSATION_PER_MINUTE`
- Los máximos de demora permitidos para cada tipo de vuelo están registrados como las constantes `MAX_LM_DELAY_ALLOWED` y `MAX_LAYOVER_DELAY_ALLOWED`

Finalmente modificar el archivo **`main.c`** para que se muestre el costo total para el día (recordar que esto se calcula con los vuelos que ocurren hasta las 6pm incluido). Cada archivo de ejemplo incluye en el nombre la cantidad esperada.