Algoritmos y Estructuras de Datos II

Laboratorio - 10/04/2025

Laboratorio 3: Tipos de Datos

- Revisión 2024: Marco Rocchietti

- Revisión 2025: Franco Luque

Código

lab03-kickstart.tar.gz

Objetivos

1. Ejercitar la resolución de problemas
2. Uso de arreglos multidimensionales y tipos **`enum`**
3. Uso de arreglos con elementos de tipo **`struct`**
4. Uso de redirección de **`stdout`** por línea de comandos
5. Lectura robusta de archivos

Requerimientos

1. Compilar con los flags de la materia:
   ```
   $ gcc -Wall -Wextra -pedantic -std=c99 …
   ```
2. Seguir las guías de estilo
3. Prohibido usar break, continue y goto!!
4. Prohibido usar return a la mitad de una función.

Recursos

Recursos generales:

- Videos del Laboratorio en el aula virtual
- Documentación en el aula virtual
- Estilo de codificación:
  - Guía de estilo para la programación en C
  - Consejos de Estilo de Programación en C

Recursos específicos:

- Teóricos:
  - Tipos Concretos
- Prácticos:
  - Práctico 2 - Parte 1
- Redirección de output

Ejercicio 1: Arreglos Multidimensionales y Estructuras

En el directorio del ejercicio se encuentran los siguientes archivos:

Archivo	*Descripción*
`main.c`	Contiene la función principal del programa
`weather.h`	Declaraciones relativas a la estructura de los datos climáticos y de funciones de carga y escritura de datos.
`weather.c`	Implementaciones **incompletas** de las funciones
`weather_table.h`	Declaraciones / prototipos de las funciones que manejan la tabla del clima
`weather_table.c`	Implementaciones **incompletas** de las funciones que manejan la tabla

Parte A: Carga de datos

Abrir el archivo **`input/weather_cordoba.in`** para ver cómo se estructuran los datos climáticos. Cada línea contiene las mediciones realizadas en un día. Las **primeras tres columnas** corresponden al *año*, *mes* y *día* de las mediciones. Las **restantes seis** columnas son la *temperatura media*, la *máxima*, la *mínima*, la *presión atmosférica*, la *humedad* y las *precipitaciones* medidas ese día.

Las temperaturas se midieron en grados centígrados (ºC) pero para evitar los números reales los grados están expresados en décimas (e.g. 15.2ºC está representado por 152 décimas). La presión (medida en *hectopascales*) también ha sido multiplicada por 10 y las precipitaciones por 100 (o sea que están expresadas en centésimas de milímetro). Esto permite representar todos los datos con números enteros. Cabe aclarar que para completar el ejercicio **no es necesario multiplicar ni dividir estos valores**, esta información es sólo para ayudar a la comprensión de los datos que se manejan.

La primera tarea consiste en completar el procedimiento de carga de datos en el archivo **`weather_table.c`**. También se debe completar **`weather.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. Como guía se puede revisar el archivo **`array_helpers.c`** provisto por la cátedra en el *laboratorio 1*.

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

```
$ gcc -Wall -Wextra -pedantic -std=c99 -c weather_table.c weather.c main.c
$ gcc -Wall -Wextra -pedantic -std=c99 weather_table.o weather.o main.o -o weather
```

y luego ejecutar con el comando:

```
$ ./weather ../input/weather_cordoba.in > weather_cordoba.out
```

En la línea anterior, **`../input/weather córdoba.in`** es el parámetro que se le pasa a nuestro programa **`weather`** (el archivo a procesar) y la parte **`> weather_cordoba.out`** hace que la salida del programa, en vez de mostrarse por la consola, se escriba en el archivo **`weather_cordoba.out.`** El archivo de salida será creado cuando comience la ejecución del programa (si **`weather_cordoba.out`** ya existía va a ser

reemplazado).

Si no hubo ningún error, ahora se puede comparar la entrada con la salida:

```
$ diff ../input/weather_cordoba.in weather_cordoba.out
```

El programa **`diff`** (que ya viene instalado en linux) realiza una comparación de ambos archivos y sólo muestra las líneas que difieren. Si esto último no arroja ninguna diferencia, significa que tu carga funciona correctamente.

Parte B: Análisis de los datos

Construir una librería **`weather_utils`** que conste de los siguientes archivos:

- **`weather_utils.c`**
- **`weather_utils.h`**

La librería debe proveer tres funciones:

1. Una función que obtenga la menor temperatura mínima histórica registrada en la ciudad de Córdoba según los datos del arreglo **(práctico 2.1, ejercicio 2.a)**.

2. Un "procedimiento" que registre para cada año entre 1980 y 2016 la mayor temperatura máxima registrada durante ese año **(práctico 2.1, ejercicio 2.b)**.

   

   El procedimiento debe tomar como parámetro un arreglo que almacenará los resultados obtenidos.

   a. Implementar un procedimiento que registre para cada año entre 1980 y 2016 el mes de ese año en que se registró la mayor cantidad mensual de precipitaciones (campo **`rainfall`**) **(práctico 2.1, ejercicio 2.c)**.

3. Finalmente modificar el archivo **`main.c`** para llamar a todas las funciones e imprimir los resultados obtenidos.

Ayudas:

- Para el procedimiento del *ítem 2* se debería hacer algo parecido a lo siguiente:

```
void procedimiento(WeatherTable a, int output[YEARS]) {
   ⋮
  for (unsigned int year = 0; year < YEARS; year++) {
     ⋮
    output[year] = ... // la mayor temperatura máxima del año  'year' + 1980
     ⋮
  }
}
```

- Para el procedimiento del *ítem 3*:
  - Ver estas filminas: Práctico 2.1: Ejercicio 2
  - Definir una o más funciones auxiliares. Por ejemplo definir **`sum_month_rainfall`**, que dado

un año y un mes, devuelve la suma de las precipitaciones de todos los días para ese mes de ese año.

- Para los ítems 2 y 3, chequear que los resultados obtenidos sean iguales a los resultados esperados de la tabla que se muestra a continuación.

Resultados esperados

Año	**ítem 2:** máxima temperatura en un día (en grados * 10)	**ítem 3:** mes de máxima lluvia
1980	380	2
1981	350	1
1982	362	3
1983	384	2
1984	363	1
1985	366	1
1986	397	12
1987	391	12
1988	394	1
1989	386	12
1990	374	1
1991	376	12
1992	337	12
1993	386	1
1994	398	1
1995	408	2
1996	368	11
1997	370	12
1998	380	2
1999	346	12
2000	370	3
2001	363	3
2002	410	1
2003	400	2
2004	393	12
2005	365	1
2006	395	11
2007	374	9
2008	390	2
2009	400	3
2010	394	3

2011	424	2
2012	408	2
2013	400	2
2014	390	2
2015	365	1
2016	380	2