3. Arreglos

Nota: Los siguientes ejercicios introductorios se refieren a programas ANSI-C no modularizados (autocontenidos dentro de la función `main()`), salvo en los ejercicios indicados donde se requiera el uso de funciones sencillas.

3.1. Arreglos y Cadenas de caracteres (*strings*)

1) Escribir un programa que lea una cadena de caracteres ingresada por `stdin` y la imprima por pantalla. Para ello utilizar las funciones de biblioteca `scanf()` con formato `"%s"`, `gets()` y `fgets()`. Comparar los resultados y explicar la conveniencia de utilizar cada una de ellas, y bajo qué circunstancias.

2) Suponga que el gerente de una PyME le pide que escriba un programa que calcule el salario semanal de un trabajador, de acuerdo con las siguientes condiciones: un empleado ingresará el nombre del trabajador, el número de horas que ha trabajado y el nivel salarial que tiene el trabajador. El programa deberá calcular el impuesto de Hacienda (se le retiene un 20 % del salario bruto) y el impuesto de seguridad social (un 8 % del salario bruto).

El programa deberá mostrar en líneas separadas, la siguiente información: el nombre del trabajador, el salario bruto, la cantidad retenida para el pago del impuesto de Hacienda, la cantidad correspondiente al pago del impuesto de la seguridad social y el salario neto del trabajador.

3) Suponga que trabaja en un videoclub. El encargado quiere que le escriba un programa que calcule el recargo que tienen que pagar los clientes cuando se retrasan en la devolución de películas de acuerdo con las siguientes normas: el alquiler de los videos cuesta 2 pesos al día, que se pagan en el momento de alquilarlos. El periodo de alquiler es de un día. El recargo por retraso es de 1 peso al día y se abonará al devolver la película. Cuando el cliente entregue la película, un empleado introducirá los siguientes datos: nombre del cliente, título de la película y número de días de retraso (que pueden ser cero). El programa deberá mostrar la siguiente información en líneas separadas: el nombre del cliente, el título de la película y el recargo por retraso.

4) Escribir un fragmento de código que determine si una cadena de caracteres está vacía o no, de dos formas distintas. Utilizar el carácter `NUL` para una de ellas, y la función de biblioteca `strcmp()` para la otra.

5) Escribir un fragmento de código que inicialice un arreglo de caracteres con una cadena cualquiera y la imprima por `stdout`.

6) Escribir un programa que inicialice arreglo de caracteres con una cadena cualquiera y la imprima en orden inverso por `stdout`.

7) a) Definir un tipo enumerativo `tformato` compuesto por los símbolos `MAYUSCULAS` y `MINUSCULAS`.

b) Escribir un programa que lea de `stdin` una cadena de caracteres y la convierta paramétricamente a minúsculas o mayúsculas, de acuerdo a una opción ingresada por el usuario a través del `stdin`, y muestre el resultado de la conversión por `stdout`. Usar tipos enumerativos para la decodificación de la selección ingresada por el usuario.

8) Escribir un programa que lea de `stdin` una cadena de caracteres, un carácter *viejo*, un carácter *nuevo*, y reemplace en la cadena todas las apariciones del caracter *viejo* por el carácter *nuevo*.

9) a) Escribir una función que dada una cadena de caracteres devuelva un booleano que indique si la misma es palíndromo o no. Considerar el caso de longitudes de cadena par e impar.

b) Escribir un programa que lea una cadena del `stdin` e informe si es palíndromo o no por el `stdout`, haciendo uso de la función programada previamente.

10) Indicar si el siguiente código es correcto o no, justificando debidamente:

```
1 char a[] = "hola";
2 char b[] = "mundo";
3 if(a==b) printf("Son iguales.");
4 else printf("Son distintas.");
```

11) Escribir un programa que lea dos números del `stdin` en formato de cadena de caracteres, las convierta a números de un tipo determinado y los compare, mostrando por `stdout` los resultados de la comparación. Para ello utilizar las funciones `atoi()` y `atof()` de la biblioteca estándar. ¿Qué puede concluir del uso de estas funciones? ¿Qué desventajas presentan?

12) Escribir un programa que lea del `stdin` un número flotante (`float`) y lo convierta a una cadena de caracteres, en forma inversa a la función `atof()`. ¿Existe la función de biblioteca `ftoa()`? Sugerencia: Puede utilizar la función de biblioteca `sprintf()`.

13) Idem para un número entero (`int`).

14) Escribir un programa que lea un número entero positivo del `stdin` y guarde su valor en base octal sobre una cadena de caracteres, mostrando el resultado de la conversión por `stdout`. Sugerencia: utilizar la función de biblioteca `sprintf()`.

15) Idem para base hexadecimal.

16) Idem para base binaria. ¿Existe en lenguaje C un formato de impresión en base binaria?

17) Escribir un programa que acepte un número seguido de un espacio y luego una letra. Si la letra que sigue al número es una *f*, el programa deberá manejar el número introducido como una temperatura en grados Fahrenheit, convertirla en grados Celsius e imprimir un mensaje adecuado de salida. Si la letra que sigue al número es una *c*, el programa deberá tratar al número como una temperatura en grados Celsius, convertirla a grados Fahrenheit, e imprimir un mensaje adecuado de salida. Si la letra no es ni una *f* ni una *c*, el programa deberá imprimir un mensaje que diga que los datos son incorrectos y terminar.

Reutilizar la función programada para el ejercicio 14 de la guía de tipos enumerativos.

Nota: En los siguientes ejercicios no se pide invocar a funciones de biblioteca ya existentes. Se pide implementar códigos que resuelvan los problemas planteados.

3.2. Manejo de cadenas de caracteres

18) Escribir un programa que dada una cadena de caracteres imprima su longitud por pantalla. Para ello implemente primero una función que reciba una cadena de caracteres, calcule su longitud y la devuelva (implementación de la función `strlen()` de la biblioteca `<string.h>`).

19) Escribir un programa que dada una cadena de caracteres y un arreglo de caracteres con espacio suficiente, copie la cadena en el arreglo, terminando la cadena con el caracter `'\0'` (implementación de la función `strcpy()` de la biblioteca `<string.h>`).

20) Escribir un programa que dadas dos cadenas de caracteres realice la concatenación de una sobre la otra, terminando la cadena con el caracter `'\0'` (implementación de la función `strcat()` de la biblioteca `<string.h>`).

21) Escribir una función que dadas dos cadenas de caracteres, realice una comparación lexicográfica de las mismas, devolviendo valores positivos, cero y negativos, según corresponda (implementación de la función `strcmp()` de la biblioteca `<string.h>`). Utilice dicha función.

22) Escribir un programa que convierta una cadena de caracteres a mayúsculas (función `strlwr()`).

23) Escribir un programa que convierta una cadena de caracteres a minúsculas (función `strupper()`).

24) Escribir una función que dadas dos cadenas de caracteres, compare lexicográficamente los primeros n caracteres, devolviendo un valor positivo, cero o negativo, según corresponda (función `strncmp()` de la biblioteca `<string.h>`). Utilice dicha función.

25) Escribir un programa que dada una cadena de caracteres y un arreglo de caracteres con espacio suficiente, copie los primeros n caracteres de la cadena sobre el arreglo, sin terminar la cadena con el caracter nulo. (Función `strncpy()` de la biblioteca `<string.h>`.)

26) Escribir un programa que lea de `stdin` una cadena de caracteres que comienza con espacios en blanco, y los elimine desplazando los caracteres útiles hacia la izquierda. (Operación *left-trim.*)

27) Escribir un programa que lea de `stdin` una cadena de caracteres que finaliza con espacios en blanco, y los elimine desplazando los caracteres útiles hacia la izquierda. (Operación *right-trim.*)

28) Escribir un programa que reciba una cadena de caracteres y un número entero no negativo n y la ajuste al margen derecho, dentro de un espacio de n caracteres.

29) Idem para margen izquierdo.

30) Escribir una función que reciba dos cadenas de caracteres denominadas `s1` y `s2` respectivamente, y verifique la existencia de la cadena s2 como subcadena integrante de la s1, imprimiendo el resultado de la búsqueda por `stdout` (ver función `strstr()` de la biblioteca `<string.h>`). Utilícela.

31) Escribir un programa que lea dos cadenas de caracteres y un número. El programa debe insertar la segunda cadena en la primera, a partir del carácter de la primera cadena que está en la posición indicada por el número.

Nota: De ahora en más, cada vez que se requiera una entrada numérica desde el `stdin` la misma debe realizarse leyendo primero sobre una cadena de caracteres y luego convirtiendo al tipo de datos deseado.

3.3. Arreglos unidimensionales

32) Escribir una función que dado un vector de un tipo determinado (`float`, `double` o `int`) y su longitud, imprima cada uno de los elementos del mismo por `stdout`.

En los ejercicios subsiguientes tendrá varias oportunidades para hacer uso de ella.

33) a) Escribir una función que dado un arreglo de `int` y su longitud, devuelva la suma de los mismos.

b) Escribir un programa que permita ingresar n números enteros en un arreglo llamado `numeros` calcule la suma total de los números en el arreglo y que despliegue el conjunto de números así como la suma de los mismos.

34) Escribir un programa que permita calcular el promedio aritmético de una serie de valores ingresados por teclado, almacenando la totalidad de los valores leídos. A tal efecto debe ingresarse primero la cantidad esperada de elementos a promediar, y luego el lote de datos de a uno por vez. La media se calcula según la expresión:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i.$$

Finalizado el ingreso de datos, mostrar el resultado por `stdout` con 3 decimales. Realizar todas las validaciones que considere necesarias. ¿Es necesario almacenar todos los datos del lote para poder calcular la media aritmética?

35) Modificar el programa del ejercicio anterior para calcular además el desvío estándar del conjunto de datos leídos. Luego mostrar por `stdout` el desvío estándar con 2 decimales. El desvío estándar de un lote muestral se calcula de acuerdo a la expresión:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2}.$$

¿Qué se puede concluir sobre el código escrito para calcular la media aritmética?

36) En relación a los dos ejercicios anteriores, se observa facilmente que no es necesario almacenar la totalidad de los datos de un lote para realizar el cálculo de la media aritmétrica. ¿Es necesario tener almacenado forzosamente todo el lote de datos, para poder calcular el desvío?

Demuestre matemáticamente que no es necesario tener almacenada la totalidad de los datos para obtener estos dos estimadores. Puede pensarse este problema como la lectura de una secuencia de valores, recalculando la media en cada iteración (Actualización de la Media). Lo mismo sucede con el cálculo del desvío estándar (problema conocido como Actualización del Desvío Estándar, y que es además función de la Media actualizada).

37) a) Escribir una función que dado un vector de un tipo determinado (`float`, `double` o `int`), su longitud y un tipo enumerativo, busque el máximo o el mínimo elemento (según lo indicado en el enumerativo) y devuelva su posición en el vector.

b) Escribir un programa que cargue en un vector de un tipo determinado valores leídos del `stdin` e informe cuanto valen el máximo y el mínimo del lote ingresado.

38) Escribir un programa que genere n números aleatorios en un vector de un tipo determinado (`float`, `double` o `int`). Mostrar su contenido por el `stdout`.

39) Escribir una función que dado un vector de números y su longitud, devuelva un valor que informe si dicho vector está ordenado o no.

40) Escribir un programa que dados dos vectores ordenados, genere un vector ordenado con los elementos de los otros dos vectores. ¿Qué relación debe haber entre las longitudes de los tres vectores para poder realizar esta operación?

41) Escribir un programa que muestree una forma de onda determinada y almacene las n muestras en un vector de `doubles`. La forma de onda es una función que responde a la expresión:

$$v(t) = A \sin(2\pi f t + \phi),$$

en donde:

t: Variable independiente (tiempo)

f: Frecuencia de la onda senoidal, en Hz.

ϕ: Fase inicial en radianes.

A: Amplitud pico de la onda senoidal.

(Implemente dicha función con todos sus parámetros para ayudarse en la carga del vector.)

Luego de terminado el proceso mostrar el contenido de las muestras por `stdout` con 4 decimales.

¿Cómo haría para volcarlas en un archivo de texto? (Redireccionar la salida por `stdout` a un archivo). Recuperar los resultados con una planilla de cálculo.

42) Se requiere un programa que muestree una función polinómica de grado g.

a) Escribir una función que reciba los coeficientes del polinomio, su grado y la variable independiente x, y retorne el polinomio evaluado en ese punto.
Sugerencia: Utilizar la Regla de Horner para hacer eficientemente este cálculo.

b) Escribir un programa que muetree un polinomio. Para ello debe almacenar n muestras en un vector de `doubles` y luego imprimir en forma tabular el contenido de las mismas por `stdout` con 3 decimales.

c) Modificar el problema anterior para volcar las muestras sobre un archivo de texto, y graficarlas con una planilla de cálculo.

43) Se tienen los siguientes juegos de caracteres:

UTF-8: Alfabeto "universal" en el cual los símbolos extendidos del castellano se representan según dos bytes.

ISO-8859-1: Alfabeto latino occidental en el cual los símbolos se representan dentro de un byte.

Entidades HTML: Una representación como cadenas ASCII de los símbolos extendidos de las codificaciones ISO-8859.

Y la siguiente tabla de códigos del idioma castellano:

Carácter	UTF-8	ISO-8859-1	Entidad
á	c3 a1	e1	á
é	c3 a9	e9	é
í	c3 ad	ed	í
ó	c3 b3	f3	ó
ú	c3 ba	fa	ú
Á	c3 81	c1	Á
É	c3 89	c9	É
Í	c3 8d	cd	Í
Ó	c3 93	d3	Ó
Ú	c3 9a	da	Ú
ü	c3 bc	fc	ü
Ü	c3 9c	dc	Ü
ñ	c3 b1	f1	ñ
Ñ	c3 91	d1	Ñ
¿	c2 bf	bf	¿
¡	c2 a1	a1	¡

Se pide:

a) Escribir un programa que convierta textos en UTF-8 a ISO-8859-1.

b) Escribir un programa que convierta textos en ISO-8859-1 a UTF-8.

c) Escribir un programa que convierta textos en UTF-8 a entidades HTML.

¿Dónde se suelen utilizar las entidades HTML? Proponga cinco posibles aplicaciones para estos programas (en realidad la conversión se suele hacer en un módulo dedicado).

44) Escribir un programa que permita ingresar los siguientes valores en un array llamado `volts`:

$$10{,}95, 16{,}32, 12{,}15, 8{,}22, 15{,}98, 26{,}22, 13{,}54, 6{,}45, 17{,}59.$$

Después de ingresar los valores, el programa debe desplegarlos de la siguiente forma:

```
10.95 16.32 12.15
8.22 15.98 26.22
13.54 6.45 17.59
```

Parametrizar adecuadamente el software desarrollado.

45) Rehacer el ejercicio 16) de la Guía N°1, utilizando vectores.

46) Dé un ejemplo de uso de arreglo de caracteres que no corresponda a un *string*.

3.4. Arreglos multidimensionales: Arreglos de cadenas de caracteres

47) Explicar la siguiente declaración: `char cadd[3][10] = { "dia", "mes", "anio"};`

48) a) Defina un tipo enumerativo `mes_t` con 12 símbolos que representen a cada uno de los meses del año.

b) Declare un arreglo de cadenas de caracteres con los nombres de los 12 meses del año.

c) Escriba una función que a partir del contenido de una variable de tipo `mes_t`, imprima por pantalla la descripción del mes (traducción).

d) Idem 48*c*) pero guardando sobre una cadena de caracteres la denominación del mes, en formato:

`MM <denominación>`

para luego imprimirla.

Sugerencia: utilizar la función de biblioteca `sprintf()`).

49) Modificar el ejercicio anterior para incluir un diccionario de meses, que permita presentar los nombres de los meses en español, inglés, y un segundo idioma extranjero. La decisión del idioma a utilizar será tomada en tiempo de compilación.

Sugerencia: utilizar compilación condicional.

50) Idem para los días de la semana.

3.5. Arreglos multidimensionales: Matrices

51) Escribir un programa que inicialice una matriz cuadrada de $n \times n$ elementos de tipo `double`, para $n = 4$, y calcule la traza de la matriz, mostrando el resultado por `stdout`.

52) Idem ejercicio anterior, pero leyendo interactivamente por `stdin` cada uno de los valores de la matriz.

53) Escriba un programa que calcule el determinante de una matriz de 2×2 y de 3×3 números reales.

54) Escribir un programa que cargue una matriz de enteros de $n \times m$ y obtenga la transpuesta de dicha matriz, mostrando el contenido de la matriz resultante por `stdout`.

55) Escribir un programa que cargue una matriz cuadrada de enteros de $n \times n$ a partir de los datos leídos por `stdin` y determine si es una matriz simétrica o no, informando los resultados por `stdout`.

56) Escribir un programa que genere dos matrices de dimensiones $n \times m$ y $m \times p$ a partir de los valores leídos en forma interactiva del `stdin`, y realice el producto de las dos matrices, mostrando la matriz resultante por `stdout`.

57) Escribir un programa que inicialize un arreglo bidimensional de números enteros, y encuentre y despliegue el máximo valor contenido en el mismo.