Práctico 3:
Arreglos en C

La teoría general para este práctico puede consultarse en los capítulos 6 y 7 de las Notas de Clase de la cátedra.

1- Hacer un programa que *ingrese* un número variable de caracteres en un arreglo y luego los *imprima*. Se deberá controlar que la cantidad ingresada sea menor o igual que la capacidad del arreglo. Para esto se deberán definir dos funciones: una que permita entrar una cantidad variable de caracteres en el arreglo, y otra que imprima los caracteres ingresados en el arreglo en formato de tabla, mostrando la posición y al lado el valor.

2- Hacer un programa que *busque* en un arreglo de números enteros un número dado, informando por pantalla si el número se encuentra o no en el arreglo. Para esto se deberán definir y usar (o re usar) las siguientes funciones: (a) una función que permita ingresar una cantidad variable de números en el arreglo; (b) una función que busca un número dado en un arreglo de enteros.

3- Hacer un programa que busque el *menor* número entero en un arreglo de enteros.
Para esto se deberán definir y usar (o re usar) las siguientes funciones:
(a) una función que permita ingresar una cantidad variable de números en el arreglo y
(b) una función que busca el menor número en un arreglo de enteros.

4- Hacer un programa que permita insertar un caracter en un arreglo de caracteres en una posición dada. Se inserta en el lugar del arreglo que el usuario del programa solicite y debe hacerse lugar al nuevo caracter, desplazando todos los caracteres una posición en el arreglo, perdiéndose el último carácter del arreglo. El programa deberá definir y usar (o reusar) una función para ingresar los datos, otra para hacer la inserción y otra para mostrar el arreglo antes y después de la inserción.

5- Hacer un programa que permita borrar un caracter en un arreglo de caracteres. Se borra en el lugar del arreglo que el usuario del programa solicite y deben desplazarse los caracteres una posición desde la posición dada hasta la posición final del arreglo. El programa deberá definir y usar (o reusar) una función para ingresar los datos, otra para hacer la supresión y otra para mostrar el arreglo antes y después de la misma.

6- Hacer un programa que dado un arreglo de números reales los *ordene* de menor a mayor. El programa deberá definir y usar (o reusar) una función para ingresar los datos, otra para ordenar el arreglo y otra para mostrar el arreglo antes y después del ordenamiento.

7- Hacer un programa que permita entrar una cantidad variable de cadenas de caracteres (strings) en un arreglo, y luego las *imprima*. Cada cadena puede tener, a su vez, un número variable de caracteres. El programa deberá definir y usar (o reusar) una función para ingresar los datos y otra para imprimir el arreglo.
Ayuda: recordar que los strings en C son arreglos de caracteres cuyo último elemento es `'\0'`, que se pueden leer y escribir en una sola operación de lectura/escritura.

8- Hacer un programa que dado pares de strings (a) indique cual es el menor (b) almacene el mayor en una variable (c) muestre la cantidad de caracteres de cada string (d) permitir al usuario ingresar 3 (o más) pares de Strings de manera que una vez finalizada la ejecución del programa muestre el String mayor de todos. Utilice la librería String.h

9- Hacer un programa que permita guardar las notas de las 13 materias que cursa un alumno de secundario en un arreglo. Las notas pueden ser: S=(10,9,8), A= (7,6,5,4), R=(3,2,1) (Superó, Aprobó, Reprobó). Para esto deberá contar con las siguientes funciones: CargaNota (carga las notas del alumno), MuestraNota (muestra las notas), Aplazos (la función deberá retornar la cantidad de R que tiene el alumno, si la cantidad es mayor o igual 7 se deberá mostrar por pantalla un mensaje: 'Alta Posibilidad de Repetir'.

10- Hacer un programa en C, que pida al usuario que ingrese el valor del dólar compra /venta de la cotización publicada por distintos bancos (10) en la página de DOLARHOY.COM. Utilice una función que tome esos valores y nos devuelva por pantalla el valor promedio de dólar compra/venta.

11- Reutilice la función del ejercicio 8 del práctico 2 de manera tal que se almacenen los horarios de ingreso y egreso de los empleados de la empresa. Utilice dos arreglos uno para horario de ingreso y otro para egreso.
Cada posición del arreglo identifica al empleado, es decir en la posición 0 está el horario de ingreso o egreso según corresponda del empleado 4000, posición 1 empleado 4001, posición 2 empleado 4002 y así sucesivamente.
Agregue además la opción que permita imprimir por pantalla, utilizando funciones, la cantidad de horas trabajadas del empleado obtenida con los datos almacenados.

12- Hacer un programa que permita a un comercio registrar los precios de 10 productos, para esto deberá guardar en un arreglo los precios de lista de los productos.
Además del arreglo de precios se debe contar con un segundo arreglo que registre los precios de promoción. La promoción consiste en precio de lista con un 15% de descuento. Para esto se deberá contar con las siguientes funciones: CargaPrecio (carga los precios de lista), CargaPromo (la función debe cargar el segundo arreglo, esto se logra leyendo del primer arreglo el precio de lista se le aplica el descuento y ése valor se guarda en el segundo arreglo), MuestraPrecios (la función muestra por pantalla tanto el precio de lista como el de promoción).

Ejercicios Complementarios:

1- Completar cada uno de los siguientes enunciados:
a. Los elementos de un arreglo se encuentran relacionados por el hecho de que tienen el mismo _____.
b. El valor usado para seleccionar a un elemento particular de un arreglo se conoce como _____.

c. Se debería usar un(a) _____ para especificar el tamaño de un arreglo porque hace al programa más mantenible y escalable.
d. Un arreglo que usa dos suscriptos o índices se lo conoce como un arreglo _____.

e. Un arreglo declarado en C int a[M][N] contiene _____ elementos en la

primera dimensión, _____ elementos en la segunda dimensión y

_____ elementos en total.
f. El segundo elemento de un arreglo a en C es seleccionado como
_____.
g. En C, para seleccionar un elemento de un arreglo d que se encuentra en la posición

tres de la primera dimensión y en la uno de la segunda usamos _____.

2- Indicar si son verdaderas o falsas las siguientes afirmaciones:
a. El tipo base de los arreglos en C es heterogéneo, es decir, se pueden almacenar valores de distintos tipos.
b. El suscripto de un arreglo en C puede ser de tipo double.
c. Si hay un número menor de inicializadores en una lista de inicializadores que el número de elementos en el arreglo, C automáticamente inicializa los restantes elementos con el último valor de la lista de inicializadores.
d. Es un error si una lista de inicializadores contiene más inicializadores que elementos tiene el arreglo.
e. Un elemento de arreglo que es pasado a una función f como un parámetro real de la forma f(a[i]) y es modificado en la función f se modificará también en la función que hace la invocación a f.
f. En C, cuando en una función uno de sus parámetros formales es un arreglo unidimensional, en la definición de la función se debe colocar su tipo base, nombre y tamaño del arreglo.
g. En C, cuando en una función uno de sus parámetros formales es un arreglo multidimensional, en la definición de la función se debe colocar su tipo base, nombre y tamaño de todas las dimensiones del arreglo excepto la primera.

3- Responder los siguientes puntos usando lo ya resuelto en los puntos anteriores, siempre que sea posible:
a. Definir un arreglo de diez elementos de tipo float llamado números e inicializar los elementos con los valores 0.0, 1.1, 2.2, ..., 9.9. Asumir que la constante simbólica SIZE ha sido definida como 10.
b. Definir un puntero, nPtr, que apunte a un objeto de tipo float.
c. Imprimir los elementos del arreglo números usando notación de suscriptos de arreglos. Usar una sentencia for y asumir que la variable de control i entera ha sido definida.
d. Dar dos sentencias alternativas para asignar la dirección base del arreglo números en el puntero nPtr.
e. Imprimir los elementos del arreglo números usando notación de punteros con el puntero nPtr.
f. Imprimir los elementos del arreglo números usando notación de punteros con el nombre del arreglo como puntero.
g. Imprimir los elementos del arreglo números usando notación de suscriptos con el puntero nPtr.
h. Referenciar al elemento 4 del arreglo números usando notación de suscriptos, notación de puntero con el nombre del arreglo como puntero, notación de suscripto con el puntero nPtr y notación de puntero con el puntero nPtr.
i. Asumir que nPtr apunta al comienzo del arreglo números. ¿Qué dirección es referenciada por nPtr + 8? ¿Qué valor se encuentra almacenado en esa dirección?
j. Asumir que nPtr apunta a numeros[5]. ¿Qué dirección es referenciada por nPtr - = 4? ¿Qué valor se encuentra almacenado en esa dirección?
4- Encontrar el error en cada uno de los siguientes trozos de programa. Asumir:
```
int *zPtr;
int *aPtr = NULL;
void *sPtr = NULL;
```

```
int number, i;
int z[5] = {1, 2, 3, 4, 5 };
sPtr = z;
```

a. ++zPtr;
b. /* usa el puntero zPtr para obtener el primer valor de un arreglo; asumir zPtr inicializado con la dirección base del arreglo z */
number = zPtr;
c. /* asigna el elemento 2 del arreglo (el valor 3) a number; asumir zPtr inicializado con la dirección base del arreglo z */
number = *zPtr[2];
d. /* imprime el arreglo z completo; asumir zPtr inicializado con la dirección base del arreglo z */
for (i = 0 ; i <= 5; i++)
printf("%d", zPtr[i]);
e. /* asigna el valor apuntado por sPtr a number */
number = *sPtr;
f. ++z;