

Arreglos

1. Responder verdadero o falso. En caso de respuestas falsas, explique el por qué.
 - a. Un arreglo puede almacenar diferentes tipos de valores.
 - b. En la expresión `a[i]`, siendo `a` un arreglo, el subíndice `i` puede ser de tipo `double`.
 - c. Si hay menos inicializadores en la lista de inicializadores del arreglo que elementos en el arreglo, C automáticamente inicializa los elementos restantes con el último valor en la lista de inicializadores.
 - d. Es un error si una lista de inicializadores contiene más inicializadores que la cantidad de elementos que hay en el arreglo.
 - e. Si se pasa un elemento a una función, en la forma `a[i]`, y se modifica ese valor en el interior de la función, el arreglo contendrá el valor modificado.

Arreglos unidimensionales

2. Escribir las definiciones siguientes:
 - a. Un arreglo de enteros con 15 elementos, todos ellos de valor 0.
 - b. Un arreglo de 7 `double`s, todos ellos de valor 3.0.
 - c. Un arreglo de `SIZE` `float`s, todos ellos de valor 0.
 - d. Un arreglo de 4 enteros con los valores: `9`, `5`, `1`, `1`.
 - e. Un arreglo de 4 caracteres con los valores: `'h'`, `'o'`, `'l'`, `'a'`.
 - f. Un arreglo de 4 caracteres con los valores: `104`, `111`, `108`, `97`.
 - g. Un arreglo con `SIZE` elementos de algún tipo (`bool`, `int`, `float`, `double`, etc) y lo inicialice con valores aleatorios.

Imprimir todos los arreglos creados y la suma de sus elementos.

3. Escribir ciclos que realicen cada una de las siguientes operaciones:
 - a. Inicializar un vector de 10 elementos con ceros.
 - b. Sumar 1 a cada uno de los 15 elementos del arreglo `vector`.
 - c. Leer del teclado y almacenar 12 valores de punto flotante en el arreglo `temp_mensuales`.
 - d. Imprimir los 5 primeros valores del vector de enteros `puntajes` en forma de columna.
 - e. Sumar un 30% a cada uno de los 20 elementos del arreglo de números `salarios`.
4. Escribir un programa que permita ingresar una cantidad fija de números, los almacene en un arreglo y luego imprima la `media` y la `varianza` del mismo.

5. Escribir un programa que pida al usuario una cantidad n de valores a generar. Luego genere n valores enteros en el intervalo $[0, 10)$ y cuente la cantidad de 0s, de 1s, de 2s, etc. Al finalizar, imprima las cantidades calculadas en forma absolutas y relativas. Realice las validaciones necesarias.

Lista 2 Ejemplo

```
Ingrese la cantidad de números a generar: 10000
0s: 1008 (0.1008)
1s: 1023 (0.1023)
2s:  967 (0.0967)
3s: 1061 (0.1061)
4s:  966 (0.0966)
5s: 1057 (0.1057)
6s:  980 (0.0980)
7s: 1010 (0.1010)
8s:  951 (0.0951)
9s:  977 (0.0977)
```

6. Rehacer el ejercicio anterior, pero imprima las ocurrencias en modo de histograma, como se muestra a continuación (note que no todos los histogramas poseen la misma cantidad de):

Elemento	Valor	Histograma
0	1008	*****
1	1023	*****
2	967	*****
3	1061	*****
4	966	*****
5	1057	*****
6	980	*****
7	1010	*****
8	951	*****
9	977	*****

En este caso, cada vale por 100 ocurrencias. Si se imprime 1 asterisco por ocurrencia, no entran en la pantalla. Para iterar de esta forma, se puede iterar de 0/1 (¿cuál?) hasta el valor y cuando el número es múltiplo de 100, se imprime un asterisco. Más eficiente es dividir el valor por la cantidad que representa cada asterisco e iterar hasta ese nuevo número.

Si la pantalla tiene 80 columnas, de las cuales se podrán utilizar 60 para el histograma ¿cuántas ocurrencias por se deben utilizar, contemplando el caso en que un número se "lleve" las 10000 ocurrencias?

7. **(todo es memoria)** Responda las siguientes preguntas y luego compruébelas utilizando la computadora.

- Dada la definición `int a[10];`: ¿cuántos elementos tiene el vector `a`? ¿Cuánta memoria ocupa el vector `a`? ¿Cuánta memoria ocupa la variable `a[2]`? ¿Dónde comienza y dónde termina la variable `a`?
- Dada la definición `float a[10];`: ¿cuántos elementos tiene el vector `a`? ¿Cuánta memoria ocupa el vector `a`? ¿Cuánta memoria ocupa la variable `a[2]`? ¿Dónde comienza y dónde termina la variable `a`?
- Dada la definición `char a[10];`: ¿cuántos elementos tiene el vector `a`? ¿Cuánta memoria ocupa el vector `a`? ¿Cuánta memoria ocupa la variable `a[2]`? ¿Dónde comienza y dónde termina la variable `a`?

Funciones con arreglos unidimensionales

! Advertencia

Recuerde que una función debe recibir el arreglo y su longitud para poder operar con el mismo.

! Nota

En **todos** los casos considere las pruebas que debe hacer, y pruebe con vectores de diferentes longitudes, por ejemplo: 0, 1, 2, 3, 10 y 11. Preste especial atención a las condiciones de borde.

8. (**matemática**) Implementar funciones que reciban un arreglo de números y su longitud y realicen las siguientes operaciones:

- completar el vector con ceros,
- completar el vector con unos,
- calcular y retornar la suma,
- calcular y retornar la media,
- calcular y retornar la varianza,
- retornar el valor máximo del arreglo,
- retornar el valor mínimo del arreglo,
- modificar los elementos del vector reemplazándolos por sus valores al cuadrado,
- modificar los elementos del arreglo reemplazando cada elemento por su signo.

$$\text{Considere } \text{sign}(x) = \begin{cases} -1 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

- modificar los elementos del arreglo reemplazándolos por las diferencias finitas de primer orden. Para hacer esto, usando como caso de ejemplo el primer y segundo elemento:

```
v[0] = v[1] - v[0];  
v[1] = v[2] - v[1];
```

9. Implementar una función que reciba 2 vectores y sus longitudes, y copie el contenido de uno en el otro, usando el siguiente prototipo:

```
bool veccpy(double dest[], size_t ldest, const double orig[], size_t lorig);
```

La función debe retornar `false` en caso de no poder completar la operación.

10. Implementar una función que reciba 2 vectores y sus longitudes, y retorne un valor booleano indicando si los vectores son iguales.
11. Implementar una función que reciba un vector, su longitud y retorne `true` si el mismo se encuentra ordenado, `false` en caso contrario. De las múltiples maneras que existen para retornar un valor booleano, elija una y justifique la respuesta.
12. **(búsqueda)** Implementar una función que reciba un vector, su longitud, un número objetivo a buscar y retorne un valor booleano indicando si el mismo se encuentra o no.
13. **(búsqueda lineal)** Implementar una función que reciba un vector, su longitud, un número objetivo a buscar y retorne la posición en la que se encuentra. ¿Qué ocurre en caso que el objetivo no se encuentre y qué se retorna? Un posible prototipo:

```
???? busqueda_lineal(const int v[], size_t n, int objetivo);
```

- ¿En qué se diferencia esta función de la del ejercicio anterior?

14. **(búsqueda binaria)** Implemente una función similar de búsqueda, utilizando búsqueda binaria. Un posible prototipo:

```
???? busqueda_binaria(const int v[], /* vector ordenado donde buscar */  
                      size_t n, /* largo del vector */  
                      int objetivo); /* elemento a buscar */
```

15. **(polinomios)** Dado un polinomio de grado n , $p(x) = a_n x^n + \dots + a_2 x^2 + a_1 x + a_0$, implemente lo siguiente:

- una función, `double polyval(const coeffs[], size_t n, double x0)`, que reciba un vector con los coeficientes a_n , el grado n del polinomio y un punto $x_0 \in \mathbb{R}$ y retorne el polinomio evaluado en dicho punto: $p(x_0) = a_n x_0^n + \dots + a_2 x_0^2 + a_1 x_0 + a_0$.
Sugerencia: buscar el método de Horner para la evaluación de polinomios.
- una función, `polyder()`, que reciba un vector con los coeficientes, el grado del polinomio y lo modifique cargando la derivada del polinomio cuyos coeficientes fueron dados.
- usando las funciones de los puntos anteriores, implemente una función que evalúe la k -ésima derivada del polinomio recibido evaluada en el punto x_0 .

- d. una función, `polypolyval()`, que dado un polinomio recibido como argumento y un vector de doubles, `double points[]`, también recibido como argumento, modifique el vector `points` guardando en cada elemento del mismo, el valor del polinomio evaluado en ese punto.

Por ejemplo, dado el polinomio $p(x) = -x^3 + x^2 + 3$ y el vector de puntos `points: {-2, 0, 2}`, luego de ejecutar `polypolyval()` con los argumentos correctos, se modifica el vector de puntos para que tenga los siguientes valores: `{15, 3, -1}`.

- e. un programa que genere una tabla con muestras de un polinomio ingresado por teclado. Se debe poder trabajar con polinomios de grado **hasta 119, no más**. El programa debe pedir al usuario, además, el valor del comienzo del intervalo de muestreo, el final y la cantidad de muestras.

Funciones con arreglos multidimensionales

! Advertencia

Recuerde que un arreglo multidimensional se debe pasar a una función indicando la longitud de todas sus dimensiones (cantidad de filas y columnas en matrices) pero además, la función que la recibe debe tener indicadas las dimensiones superiores (columnas en las matrices). Recordando el ejercicio de la matriz identidad:

Dada la variable `double arr[N][N];`, la función `identidad()` tendría el siguiente prototipo: `void identidad(double matriz[][N], size_t, size_t)`.

16. Escriba un programa que inicialice una matriz de N filas y M columnas con:

- ceros,
- unos,
- los números del 1 al NM ,
- números indicados por el usuario a través del teclado,
- números aleatorios,
- la sucesión de fibonacci,

y la muestre por pantalla.

17. Para cada uno de los casos anteriores, implemente una función que reciba y cargue la matriz con los valores adecuados.

18. (**programas con matrices**) Para cada inciso escriba un programa que lo resuelva, o escriba un programa que resuelva todos:

- cargue una matriz $A \in \mathbb{R}^{N \times N}$ y calcule su traza (suma de los elementos de la diagonal principal),
- cargue una matriz $A \in \mathbb{R}^{N \times M}$ y número y modifique la matriz sumando a cada componente el número recibido,

- c. cargue una matriz $\in \mathbb{R}^{N \times M}$ y número y modifique la matriz multiplicando a cada componente por el número recibido,
- d. cargue una matriz $A \in \mathbb{R}^{N \times M}$ y modifique dicha matriz cambiándola por su transpuesta o lea otra matriz B con las dimensiones adecuadas y la cargue con la transpuesta de A ,
- e. cargue una matriz $A \in \mathbb{R}^{N \times M}$ e indique si la misma es positiva, no-negativa, negativa o no-positiva. Una matriz es positiva (no-negativa) si cumple que todos sus elementos son mayores (mayores o iguales) que cero. Una matriz es negativa (no-positiva) si cumple que todos sus elementos son menores (menores o iguales) que cero. Por ejemplo:

- $A_1 = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ es positiva ($a_{ij} > 0 \forall i, j$),
- $A_2 = \begin{bmatrix} 1 & 0 \\ 3 & 4 \end{bmatrix}$ es no negativa ($a_{ij} \geq 0 \forall i, j$),
- $A_3 = \begin{bmatrix} -1 & -2 \\ -3 & -4 \end{bmatrix}$ es negativa ($a_{ij} < 0 \forall i, j$),
- $A_4 = \begin{bmatrix} -1 & 0 \\ -3 & -4 \end{bmatrix}$ es no positiva ($a_{ij} \leq 0 \forall i, j$),
- $A_5 = \begin{bmatrix} 1 & 0 \\ 0 & -4 \end{bmatrix}$ no cumple ninguna de las condiciones.

- f. sólo para matrices de 2×2 ó 3×3 : calcule el determinante,
- g. dadas 2 matrices $A \in \mathbb{R}^{N \times K}$ y $B \in \mathbb{R}^{K \times M}$, calcule el producto almacenando el resultado en una tercera matriz C ,
- h. retorne el máximo elemento de la matriz,
- i. retorne el máximo de la suma, en valores absolutos, de los elementos de cada columna (norma-1 matricial),
- j. retorne el máximo de la suma, en valores absolutos, de los elementos de cada fila (norma-infinito matricial).
- k. Tómese un descanso.

19. **(funciones con matrices)** Repita el ejercicio anterior, pero utilizando funciones. Agregue el siguiente inciso:

- l. Tómese dos descansos.

Gráficos Net PBM

El formato Net PBM [\[inglés\]](#) [\[español\]](#) es un formato de imágenes muy sencillo. Básicamente, una imagen Net PBM es un archivo de texto que sigue el siguiente formato:

```
P1
ANCHO ALTO
SECUENCIA de 0s y 1s
```

Por ejemplo

```
P1
20 7
00000000000000000000
01110011100110001100
01010010000010000100
01010011000010000100
01110000100010000100
00010000100010000100
01110011100111001110
```

que da como resultado (ampliado 10 veces):



20. Escriba una función que reciba una matriz, con un ancho máximo de 1050 columnas aunque lo utilizado puede ser menor, e imprima por pantalla el contenido de un archivo Net PBM. Por ejemplo, dada una matriz `canvas` definida como: `uchar canvas[1680][1050];` cargada con los datos del ejemplo anterior, imprima lo que se ve en el ejemplo anterior. Prototipo:

```
bool canvas2pbm(uchar canvas??, size_t w, size_t h);
```

21. Escriba una función que reciba una matriz y dibuje sobre ella el contenido de otra matriz, en la posición indicada a través de un par de argumentos. Prototipo:

```
bool
pbmpaste(uchar canvas??, /* matriz de 1680x1050 */
          size_t w,      /* ancho del canvas */
          size_t h,      /* alto del canvas */
          const uchar img??, /* imagen a pegar */
          size_t iw,     /* ancho de la imagen */
          size_t ih,     /* alto de la imagen */
          size_t x,      /* eje x donde pegar la imagen */
          size_t y,      /* eje y donde pegar la imagen */
          );
```

22. Escriba una función que reciba una matriz y dibuje sobre ella una línea, recibiendo las coordenadas de inicio y fin como argumentos. Prototipo:

```
bool draw_line(uchar canvas??, size_t w, size_t h, size_t x1, size_t y1, size_t x2,
               size_t y2);
```

23. Escriba una función que reciba una matriz y dibuje sobre ella un cuadrado de lado l , en la posición (i, j) . Para dibujar, debe poner unos en las posiciones requeridas.
24. Escriba una función que reciba una matriz y dibuje sobre ella una circunferencia de radio r , en la posición (i, j) .

Conway's game of LIFE

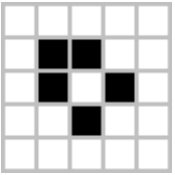
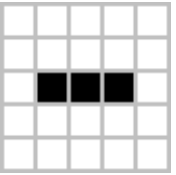
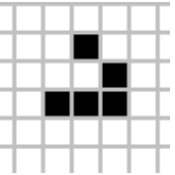
Ver [\[inglés\]](#) [\[español\]](#).

El juego de la vida de John Horton Conway es un "juego", sin jugadores, en un "tablero" o "grilla" bidimensional, donde cada celda del tablero tiene 2 estados, *viva* o *muerta*. Cada celda del *universo* interactúa con sus 8 vecinos (arriba, abajo, izquierda, derecha, arriba izquierda, etc.). Dada una configuración inicial del tablero, el juego evoluciona de un estado al otro dadas las siguientes reglas:

1. Una celda con *menos* de 2 vecinos vivos, muere de aburrimiento.
2. Una celda con 2 o 3 vecinos vivos, vive (pero está complotando).
3. Una celda con más de 3 vecinos vivos, es asesinada y muere.
4. Una celda **muerta** con *exactamente* 3 vecinos vivos, revive para vengarse.

El juego finaliza después de una cantidad fija de iteraciones o bien cuando no hay cambios de una generación a la siguiente.

Es sabido que hay ciertas estructuras "interesantes" en este juego de la vida. Por ejemplo, figuras estáticas, periódicas ("osciladores") y *spaceships*:

Estático	Oscilador	Spaceship
		
<i>Boat</i>	<i>Blinker</i>	<i>Glider</i>

25. Escribir una función que dada una matriz de un cierto ancho y alto, inicialice aleatoriamente las celdas en estados **viva** o **muerta**. Prototipo:

```
bool conways_random_init(??? grid ???, size_t, size_t);
```

Debe completar el prototipo con argumentos coherentes.

26. Escribir una función que dada una matriz, y una posición, copie un *Boat*, o *Blinker* o *Glider* en dicha posición. Prototipos:


```
bool conways_boat(??? grid ???, size_t, size_t, size_t, size_t);
bool conways_blinker(??? grid ???, size_t, size_t, size_t, size_t);
bool conways_glider(??? grid ???, size_t, size_t, size_t, size_t);
```

27. Escribir una función que reciba 2 grillas, una vieja generación y una nueva generación y aplique las reglas, grabando en nueva generación el resultado.
28. Escribir una función que reciba una grilla del juego de la vida e imprima por pantalla el estado actual del juego con el formato de un archivo Net PBM.
29. En el ejercicio anterior, si no se dió cuenta se lo comento, se podría haber reutilizado parte o todo el código desarrollado en la sección sobre los gráficos Net PBM.
 - a. ¿En dónde?
 - b. ¿Cómo se puede reutilizar dicho código?
 - c. Si las funciones desarrolladas son modificadas, pero siguen haciendo lo que dice el enunciado, sólo que lo hacen de otra forma ¿cree que sería necesario modificar el código de esta sección?
 - d. ¿Qué tipo de acoplamiento ve entre los módulos desarrollados?
30. Con todo lo visto, desarrolle una aplicación para ejecutar el juego de la vida de Conway. Es recomendable considerar ambas condiciones de finalización del juego. Considere probar la aplicación borrando la pantalla en cada iteración e imprimiendo por pantalla únicamente las celdas vivas. Para borrar la pantalla, en GNU/Linux, utilice la función `system()` de la biblioteca estándar `stdlib.h`. La misma recibe un comando a ejecutar, que es equivalente a ejecutar en la terminal. En este caso, ejecute "clear":
`system("clear");`. De todos modos, seguramente se ejecute demasiado rápido como para notar resultados sin utilizar algún otro mecanismo para parar la ejecución, como `getchar()`.