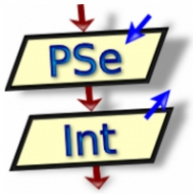
CURSO DE PROGRAMACIÓN FULL STACK

ARREGLOS CON PSEINT

MATRICES



EGG

EJERCICIOS DE APRENDIZAJE

Para cada uno de los siguientes ejercicios realizar el análisis del problema e indicar cuáles son los datos de entrada y cuáles son los datos de salida. Escribir luego el algoritmo en PSeInt haciendo uso de funciones y/o procedimientos según corresponda en cada caso.

1. Realizar un programa que rellene una matriz de 3x3 con 9 valores ingresados por el usuario y los muestre por pantalla.

2. Escribir un programa que realice la búsqueda lineal de un número entero ingresado por el usuario en una matriz de 5x5, llena de números aleatorios y devuelva por pantalla las coordenadas donde se encuentra el valor, es decir en que fila y columna se encuentra. En caso de no encontrar el valor dentro de la matriz se debe mostrar un mensaje.

3. Dada una matriz de orden n * m (donde n y m son valores ingresados por el usuario) realizar un subprograma que llene la matriz de numeros aleatorios. Despues, crearemos otro subprograma que calcule y muestre la suma de los elementos de la matriz. Mostrar la matriz y los resultados por pantalla.

4. Rellenar en un subproceso una matriz cuadrada con números aleatorios salvo en la diagonal principal, la cual debe rellenarse con ceros. Una vez llena la matriz debe generar otro subproceso para imprimir la matriz.

5. Rellenar una matriz, de 3 x 3, con una palabra de 9 de longitud, pedida por el usuario, encontrando la manera de que la frase se muestre de manera continua en la matriz.
   Por ejemplo, si tenemos la palabra habilidad, nuestra matriz se debería ver así:

H	A	B
I	L	I
D	A	D

   Nota: recordar el uso de la función Subcadena().

6. Una matriz mágica es una matriz cuadrada (tiene igual número de filas que de columnas) que tiene como propiedad especial que la suma de las filas, las columnas y las diagonales es igual. Por ejemplo:

   2 7 6
   9 5 1
   4 3 8

   En la matriz de ejemplo las sumas son siempre 15. Considere el problema de construir un algoritmo que compruebe si una matriz de datos enteros es mágica o no, y en caso de que sea mágica escribir la suma. Además, el programa deberá comprobar que los números introducidos son correctos, es decir, están entre el 1 y el 9. El usuario ingresa el tamaño de la matriz que no debe superar orden igual a 10.

EJERCICIOS DE APRENDIZAJE EXTRA

Estos van a ser ejercicios para reforzar los conocimientos previamente vistos. Estos pueden realizarse cuando hayas terminado la guía y tengas una buena base sobre lo que venimos trabajando. Además, si ya terminaste la guía y te queda tiempo libre en las mesas, podes continuar con estos ejercicios extra, recordando siempre que no es necesario que los termines para continuar con el tema siguiente. Por ultimo, recordá que la prioridad es ayudar a los compañeros de la mesa y que cuando tengas que ayudar, lo más valioso es que puedas explicar el ejercicio con la intención de que tu compañero lo comprenda, y no sólo mostrarlo. ¡Muchas gracias!

1. Realizar un programa que rellene de números aleatorios una matriz a través de un subprograma y generar otro subprograma que muestre por pantalla la matriz final.

2. Crear una matriz de orden n * m (donde n y m son valores ingresados por el usuario), llenarla con números aleatorios entre 1 y 100 y mostrar su traspuesta. NOTA: si no conoces lo que es una traspuesta, mirar el siguiente link: Matriz Traspuesta

3. Realizar un programa que cree una matriz de 5x15 y deberemos llenar la matriz de unos y ceros. Llenando el marco o la delimitación externa de la matriz de unos y la parte interna de ceros.

   Por ejemplo, nuestro matriz final debería verse así:

   ```
   111111111111111
   100000000000001
   100000000000001
   100000000000001
   111111111111111
   ```

4. Realizar un programa que calcule la multiplicación de dos matrices de enteros de 3x3. Inicialice las matrices para evitar el ingreso de datos por teclado.

5. Crear una matriz que contenga 3 columnas y la cantidad filas que decida el usuario. Las dos primeras columnas contendrán valores enteros ingresados por el usuario y en la 3 columna se deberá almacenar el resultado de sumar el número de la primera y segunda columna. Mostrar la matriz de la siguiente forma:

   3 + 5 = 8
   4 + *3* = *7*
   1 + *4* = *5*

6. Realizar un programa que permita visualizar el resultado del producto de una matriz de enteros de 3x3 por un vector de 3 elementos. Los valores de la matriz y el vector pueden inicializarse evitando así el ingreso de datos por teclado. Para conocer más acerca de cómo se realiza la multiplicación entre matrices consultar el siguiente link:

   https://es.wikibooks.org/wiki/%C3%81lgebra_Lineal/Matriz_por_vector

7. Una empresa de venta de productos por correo desea realizar una estadística de las ventas realizadas de cada uno de sus productos a lo largo de una semana. Distribuya luego 5 productos en los 5 días hábiles de la semana. Se desea conocer:

    a. Total de ventas por cada día de la semana.
    b. Total de ventas de cada producto a lo largo de la semana.
    c. El producto más vendido en cada semana.
    d. El nombre, el día de la semana y la cantidad del producto más vendido.

    El informe final tendrá un formato como el que se muestra a continuación:

	Lunes	Martes	Miércoles	Jueves	Viernes	Total producto
Producto 1						
Producto 2						
Producto 3						
Producto 4						
Producto 5						
Total semana						
Producto más vendido						

8. Una distribuidora de Nescafé tiene 4 representantes que viajan por toda la Argentina ofreciendo sus productos. Para tareas administrativas el país está dividido en cinco zonas: Norte, Sur, Este, Oeste y Centro. Mensualmente almacena sus datos y obtiene distintas estadísticas sobre el comportamiento de sus representantes en cada zona. Se desea hacer un programa que lea el monto de las ventas de los representantes en cada zona y calcule luego:

    a) el total de ventas de una zona introducida por teclado
    b) el total de ventas de un vendedor introducido por teclado en cada una de las zonas
    c) el total de ventas de todos los representantes.

EJERCICIOS COMPLEMENTARIOS INTEGRADORES

1. "Salida de un laberinto": Se trata de encontrar un camino que nos permita salir de un laberinto definido en una matriz NxN. Para movernos por el laberinto, sólo podemos pasar de una casilla a otra que sea adyacente a la primera y no esté marcada como una casilla prohibida (esto es, las casillas prohibidas determinan las paredes que forman el laberinto).

   Algoritmo recursivo:

   - Se comienza en la casilla (0,0) y se termina en la casilla (N-1, N-1)
   - Nos movemos a una celda adyacente si esto es posible.
   - Cuando llegamos a una situación en la que no podemos realizar ningún movimiento que nos lleve a una celda que no hayamos visitado ya, retrocedemos sobre nuestros pasos y buscamos un camino alternativo.

2. "Batalla naval espacial": Este juego se juega en un tablero de 4 x 4, donde las filas se identifican de la A hasta la D y las columnas del 1 al 4. En el juego participan 2 contendientes: el defensor y el atacante. Dicho juego consiste en:

   El *defensor*, ubica solo una nave nodriza triple con ciertas reglas:

   2.1) La nave debe ubicarse de tal forma que sus partes queden contiguas, ya sea horizontal o vertical, pero no es válido en forma oblicua.

   2.2) Cada una de las tres partes que compone la nave contiene un escudo de electrones medido con un valor del 1 al 9, el cual debe pedirse al usuario junto con su posición.

   A continuación, se ilustra un ejemplo de una ubicación posible:

	1	2	3	4
A				
B				
C		**4**	**7**	**1**
D				

   2.3) El atacante, indicando una coordenada del tablero (por ejemplo, C3) y una carga de protones, debe intentar acertar a la nave de su contrincante. El ataque, posee las siguientes reglas:

   a) La carga de protones asociada al ataque corresponde a un valor del 1 al 9.
   b) ¡Si el atacante no acierta en la posición, entonces el defensor informa "Espacio!".
   c) Si el atacante acierta la posición:

c.1) El ataque es “efectivo” y resta el valor de la carga protones al escudo de electrones, si y solo sí, el valor de la carga de protones es menor o igual al valor restante de electrones del escudo. En el ejemplo de ubicación anterior si el atacante indica C3 con carga 9, el ataque es “sin efecto” y no genera daño alguno. Pero si indica C3 con carga 4 el ataque es “efectivo” y el escudo de la posición queda con carga de 3 electrones.

c.2) Luego del ataque se debe indicar si fue efectivo o no, si se neutralizó o no el escudo del casillero y la suma total de electrones que resta para hundir la nave. El escudo de un casillero se neutraliza cuando llega a cero. Suponiendo que en el primer ataque se indica C3 con carga 4, se indica “Ataque efectivo – Escudo no neutralizado – Carga restante de electrones igual a 3”.

d) Cada vez que el atacante realiza un disparo resta el valor de la carga de su reactor de protones. El reactor de la nave atacante es de 40 protones. Un disparo a realizar no puede superar la carga de protones restantes.

El juego termina cuando se cumple alguna de las siguientes situaciones:

a) Gana el atacante cuando deja sin escudos a la nave nodriza y todavía le queda carga para un disparo más.
b) Gana el defensor cuando el atacante se queda sin carga en el reactor de protones.

Realice un programa que implemente la lógica del juego, iniciando con la distribución de la nave en el tablero por parte del defensor, y luego desarrollando la partida del atacante hasta la culminación del juego. El programa debe indicar quién ganó el juego.