Taller 9 semana 10

Ejercicios Pares

Ejercio 2

Inico

DIMENSION inventario[100][3]

totalProductos <- 0

opcion <- 0

REPETIR

ESCRIBIR "1. Agregar producto"

ESCRIBIR "2. Actualizar producto"

ESCRIBIR "3. Buscar producto"

ESCRIBIR "4. Eliminar producto"

ESCRIBIR "5. Mostrar inventario"

ESCRIBIR "Seleccione una opción: "

LEER opcion

SEGUN opcion

CASO 1:

SI totalProductos < LONGITUD(inventario)

ESCRIBIR "Ingrese el nombre del producto: "

inventario[totalProductos][0] <- LEER

ESCRIBIR "Ingrese el precio del producto: "

inventario[totalProductos][1] <- LEER

ESCRIBIR "Ingrese la cantidad del producto: "

inventario[totalProductos][2] <- LEER

totalProductos <- totalProductos + 1

ESCRIBIR "Producto agregado con éxito."

SINO

ESCRIBIR "No se pueden agregar más productos."

FIN SI

CASO 2:

ESCRIBIR "Ingrese el nombre del producto a actualizar: "

nombreProducto <- LEER

encontrado <- FALSO

PARA i DESDE 0 HASTA totalProductos - 1

SI inventario[i][0] != NULO Y inventario[i][0] == nombreProducto ENTONCES

ESCRIBIR "Ingrese la nueva cantidad del producto: "

inventario[i][2] <- LEER

ESCRIBIR "Producto actualizado."

encontrado <- VERDADERO

SALIR PARA

FIN SI

FIN PARA

SI NO encontrado

ESCRIBIR "Producto no encontrado en el inventario."

FIN SI

CASO 3:

ESCRIBIR "Ingrese el nombre del producto a buscar: "

nombreBuscar <- LEER

encontradoBuscar <- FALSO

PARA i DESDE 0 HASTA totalProductos - 1

SI inventario[i][0] != NULO Y inventario[i][0] == nombreBuscar ENTONCES

ESCRIBIR "Producto encontrado:"

ESCRIBIR "Nombre: " + inventario[i][0]

ESCRIBIR "Precio: " + inventario[i][1]

ESCRIBIR "Existencias: " + inventario[i][2]

encontradoBuscar <- VERDADERO

SALIR PARA

FIN SI

FIN PARA

SI NO encontradoBuscar

ESCRIBIR "Producto no encontrado en el inventario."

FIN SI

CASO 4:

ESCRIBIR "Ingrese el nombre del producto a eliminar: "

nombreEliminar <- LEER

encontradoEliminar <- FALSO

PARA i DESDE 0 HASTA totalProductos - 1

SI inventario[i][0] != NULO Y inventario[i][0] == nombreEliminar ENTONCES

inventario[i][0] <- NULO

inventario[i][1] <- NULO

inventario[i][2] <- NULO

ESCRIBIR "Producto eliminado con éxito."

encontradoEliminar <- VERDADERO

SALIR PARA

FIN SI

FIN PARA

SI NO encontradoEliminar

ESCRIBIR "Producto no encontrado en el inventario."

FIN SI

CASO 5:

ESCRIBIR "Inventario:"

ESCRIBIR "Nombre Precio Existencias"

PARA i DESDE 0 HASTA totalProductos - 1

SI inventario[i][0] != NULO ENTONCES

ESCRIBIR inventario[i][0] + " " + inventario[i][1] + " " + inventario[i][2]

FIN SI

FIN PARA

CASO POR DEFECTO:

ESCRIBIR "Opción no válida. Por favor, seleccione una opción válida."

FIN SEGUN

HASTA QUE opcion == 0

CERRAR teclado

FIN

Prueba de Escritorio

Inventario:

Nombre Precio Existencias

sardina 2.50 600

atun 2.75 450

arroz 0.60 300

Ejercicio 4

Inicio

tablero = [

[' ', ' ', ' '],

[' ', ' ', ' '],

[' ', ' ', ' ']

]

jugadorActual = 'X'

Mientras Verdadero

imprimirTablero(tablero)

hacerMovimiento(tablero, jugadorActual)

Si haGanado(tablero, jugadorActual) Entonces

imprimirTablero(tablero)

Mostrar "¡Jugador " + jugadorActual + " ha ganado!"

Romper

Sino Si esEmpate(tablero) Entonces

imprimirTablero(tablero)

Mostrar "¡Empate!"

Romper

jugadorActual = Si jugadorActual == 'X' Entonces 'O' Sino 'X'

Fin

Función imprimirTablero(tablero)

Mostrar "-------------"

Para Cada Fila en tablero

Mostrar "| " + Unir(Fila, " | ") + " |"

Mostrar "-------------"

Fin Para

Fin Función

Función hacerMovimiento(tablero, jugador)

entrada = Crear Objeto de Entrada

Hacer

Mostrar "Jugador " + jugador + ", ingresa la fila (0-2): "

fila = Convertir a Entero(entrada.ObtenerEntrada())

Mostrar "Jugador " + jugador + ", ingresa la columna (0-2): "

columna = Convertir a Entero(entrada.ObtenerEntrada())

Mientras fila < 0 o fila >= 3 o columna < 0 o columna >= 3 o tablero[fila][columna] != ' '

tablero[fila][columna] = jugador

Fin Función

Función haGanado(tablero, jugador)

Para Cada i en Rango(3)

Si (tablero[i][0] == jugador y tablero[i][1] == jugador y tablero[i][2] == jugador) o

(tablero[0][i] == jugador y tablero[1][i] == jugador y tablero[2][i] == jugador) Entonces

Devolver Verdadero

Fin Si

Fin Para

Si (tablero[0][0] == jugador y tablero[1][1] == jugador y tablero[2][2] == jugador) o

(tablero[0][2] == jugador y tablero[1][1] == jugador y tablero[2][0] == jugador) Entonces

Devolver Verdadero

Fin Si

Devolver Falso

Fin Función

Función esEmpate(tablero)

Para Cada i en Rango(3)

Para Cada j en Rango(3)

Si tablero[i][j] == ' ' Entonces

Devolver Falso

Fin Si

Fin Para

Fin Para

Devolver Verdadero

Fin Función

Fin

PRUEBA DE ESCRITORIO

-------------

| X | X | X |

-------------

| | | O |

-------------

| | O | |

-------------

Ejercicio 6

Inicio

Matriz matrizA, matrizB

Entero opcion

matrizA = generarMatrizAleatoria()

matrizB = generarMatrizAleatoria()

Hacer

Imprimir "Calculadora de Matrices"

Imprimir "Menú:"

Imprimir "1. Suma de Matrices"

Imprimir "2. Resta de Matrices"

Imprimir "3. Salir"

Imprimir "Seleccione una opción: "

Leer opcion

Según opcion Hacer

Caso 1:

mostrarMatriz("Matriz A", matrizA)

mostrarMatriz("Matriz B", matrizB)

Matriz resultadoSuma = sumarMatrices(matrizA, matrizB)

mostrarMatriz("Resultado de la Suma", resultadoSuma)

Fin Caso

Caso 2:

mostrarMatriz("Matriz A", matrizA)

mostrarMatriz("Matriz B", matrizB)

Matriz resultadoResta = restarMatrices(matrizA, matrizB)

mostrarMatriz("Resultado de la Resta", resultadoResta)

Fin Caso

Caso 3:

Imprimir "¡Hasta luego!"

Fin Caso

De Otro Modo:

Imprimir "Opción no válida. Intente de nuevo."

Fin De Otro Modo

Fin Según

Mientras opcion no sea 3

Función generarMatrizAleatoria() Retorna Matriz

Matriz matriz

Para cada fila en rango de 1 a 3 Hacer

Para cada columna en rango de 1 a 3 Hacer

matriz[fila][columna] = EnteroAleatorioEntre(-9, 9)

Fin Para

Fin Para

Retornar matriz

Fin Función

Procedimiento mostrarMatriz(nombre: Cadena, matriz: Matriz)

Imprimir "Matriz " + nombre + ":"

Para cada fila en matriz Hacer

Para cada elemento en fila Hacer

Imprimir elemento + " "

Fin Para

Imprimir NuevaLínea

Fin Para

Fin Procedimiento

Función sumarMatrices(matrizA: Matriz, matrizB: Matriz) Retorna Matriz

Matriz resultado

Para cada fila en rango de 1 a 3 Hacer

Para cada columna en rango de 1 a 3 Hacer

resultado[fila][columna] = matrizA[fila][columna] + matrizB[fila][columna]

Fin Para

Fin Para

Retornar resultado

Fin Función

Función restarMatrices(matrizA: Matriz, matrizB: Matriz) Retorna Matriz

Matriz resultado

Para cada fila en rango de 1 a 3 Hacer

Para cada columna en rango de 1 a 3 Hacer

resultado[fila][columna] = matrizA[fila][columna] - matrizB[fila][columna]

Fin Para

Fin Para

Retornar resultado

Fin Función

Función EnteroAleatorioEntre(min: Entero, max: Entero) Retorna Entero

Retornar EnteroAleatorioEnRango(max - min + 1) + min

Fin Función

Función EnteroAleatorioEnRango(rango: Entero) Retorna Entero

Retornar EnteroAleatorio() % rango

Fin Función

Fin

PRUEBA DE ESCRITORIO

Inicio

Si el usuario ingresa 1

Proceso

Matriz A:

2 -8 8

-7 -6 8

-5 -2 9

Matriz B:

-3 -4 -3

6 1 -7

2 2 2

Salida

Resultado de la Suma:

-1 -12 5

-1 -5 1

-3 0 11

Si el usuario ingresa 2

Matriz A:

2 -8 8

-7 -6 8

-5 -2 9

Matriz B:

-3 -4 -3

6 1 -7

2 2 2

Resultado de la Resta:

5 -4 11

-13 -7 15

-7 -4 7

Ejercicio 8

Inicio

Definir limMatr como entero

limMatr <- 5

Definir mtrx como matriz de enteros de tamaño [limMatr][limMatr]

Para cada fila i desde 0 hasta limMatr - 1 hacer

Para cada columna j desde 0 hasta limMatr - 1 hacer

mtrx[i][j] <- GenerarNumeroAleatorioEntre(-9, 9)

Mostrar mtrx[i][j] + "\t"

Fin Para

Mostrar nueva línea

Fin Para

Mostrar "Diagonal"

Para cada i desde 0 hasta limMatr - 1 hacer

Mostrar mtrx[i][i] + "\t"

Fin Para

Mostrar nueva línea

Mostrar "Sobre"

Para cada i desde 0 hasta limMatr - 1 hacer

Para cada j desde i + 1 hasta limMatr - 1 hacer

Mostrar mtrx[i][j] + "\t"

Fin Para

Mostrar nueva línea

Fin Para

Mostrar "Debajo"

Para cada i desde 0 hasta limMatr - 1 hacer

Para cada j desde 0 hasta i - 1 hacer

Mostrar mtrx[i][j] + "\t"

Fin Para

Mostrar nueva línea

Fin Para

Fin

Prueba de Escritorio

6 8 2 3 3

1 5 -4 5 6

3 9 7 -5 6

9 4 -7 7 -1

8 6 7 0 -4

Diagonal

6 5 7 7 -4

Sobre

8 2 3 3

-4 5 6

-5 6

-1

Debajo

1

3 9

9 4 -7

8 6 7 0

Ejercicio 10

Inicio

tamano = 4

Para i desde 1 hasta tamano Paso 1

// Imprimir espacios en blanco

Para j desde tamano - i hasta 1 Paso -1

Imprimir " "

Fin Para

Para k desde 1 hasta i Paso 1

Imprimir "$ "

Fin Para

Imprimir Nueva línea

Fin Para

Para i desde tamano - 1 hasta 1 Paso -1

Para j desde 0 hasta tamano - i - 1 Paso 1

Imprimir " "

Fin Para

Para k desde 1 hasta i Paso 1

Imprimir "$ "

Fin Para

Imprimir Nueva línea

Fin Para

Fin Algoritmo

Prueba de Escritorio

$

$ $

$ $ $

$ $ $ $

$ $ $

$ $

$