1D Arrays

Task No 01:

Create an array A of length 10 of integer values ranging from 1 to 50.

1. Find all pairs of elements whose sum is 25.
2. Find the number of elements of A which are even, and the number of elements of A which are odd.
3. Write a procedure which finds the average of the value of A.

Solution:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab01Task01

{

class Program

{

static void Main(string[] args)

{

int[] A = new int[] { 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 };

for (int i = 0; i < A.Length; i++)

{

for (int j = 0; j < A.Length; j++)

{

if (A[i] + A[j] == 25)

{

Console.WriteLine("The Sum of i : \t " + A[i] + "\t and j : \t " + A[j] + "\t is equal to 25");

}

}

}

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Even And Odd\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

int even = 0, odd = 0;

for (int i = 0; i < A.Length; i++)

{

if (A[i] % 2 == 0)

{

even++;

}

else

{

odd++;

}

}

Console.WriteLine("There are {0} Odd and {1} Even", odd, even);

int sum = 0;

//Average

Console.WriteLine("------------Average---------");

for (int i = 0; i < A.Length; i++)

{

sum += A[i];

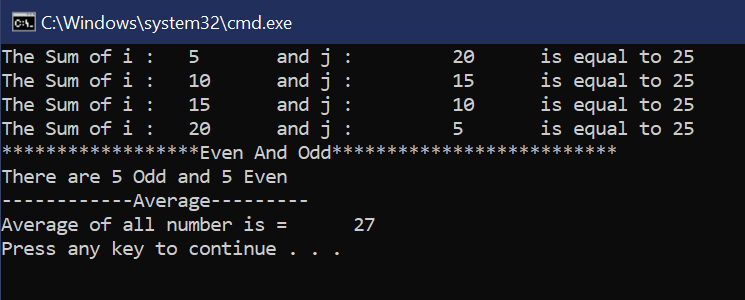
}

Console.WriteLine("Average of all number is =\t" + sum / A.Length);

}

}

Output:



Task No 02:

Write a C# program that utilizes a 1D array to implement a simple inventory management system.

1: Inventory Setup

Create an array to store inventory items and initialize the item count.

2: Main Menu Loop

Implement the main menu loop for the inventory management system. This loop will repeatedly display options to the user until they choose to exit.

3: Add Item

Implement the functionality to add an item to the inventory. Ask the user for the name of the item to add and add it to the inventory array.

4: Remove Item

Implement the functionality to remove an item from the inventory. Ask the user for the name of the item to remove and remove it from the inventory array if found.

5. Search Item

Implement the functionality to search for an item in the inventory. Ask the user for the name of the item to search for and display whether it's in the inventory or not.

6: Display Inventory

Implement the functionality to display the current items in the inventory.

7: Exit Program

Implement the functionality to exit the inventory management system when the user chooses to exit.

Solution:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab01Task02

{

class Program

{

static string[] inventory = new string[50]; // Max inventory size of 50

static int itemCount = 0;

static void Main(string[] args)

{

bool exit = false;

while (!exit)

{

Console.WriteLine("Inventory Management System");

Console.WriteLine("1: Add Item");

Console.WriteLine("2: Remove Item");

Console.WriteLine("3: Search Item");

Console.WriteLine("4: Display Inventory");

Console.WriteLine("5: Exit");

Console.Write("Select an option (1-5): ");

int choice;

if (int.TryParse(Console.ReadLine(), out choice))

{

switch (choice)

{

case 1:

AddItem();

break;

case 2:

RemoveItem();

break;

case 3:

SearchItem();

break;

case 4:

DisplayInventory();

break;

case 5:

exit = true;

break;

default:

Console.WriteLine("Invalid option. Please choose a valid option (1-5).");

break;

}

}

else

{

Console.WriteLine("Invalid input. Please enter a number (1-5).");

}

Console.WriteLine();

}

}

static void AddItem()

{

if (itemCount < inventory.Length)

{

Console.Write("Enter the name of the item to add: ");

string newItem = Console.ReadLine();

inventory[itemCount] = newItem;

itemCount++;

Console.WriteLine($"Item '{newItem}' added to inventory.");

}

else

{

Console.WriteLine("Inventory is full. Cannot add more items.");

}

}

static void RemoveItem()

{

Console.Write("Enter the name of the item to remove: ");

string itemToRemove = Console.ReadLine();

bool itemFound = false;

for (int i = 0; i < itemCount; i++)

{

if (inventory[i] == itemToRemove)

{

for (int j = i; j < itemCount - 1; j++)

{

inventory[j] = inventory[j + 1];

}

itemCount--;

itemFound = true;

Console.WriteLine($"Item '{itemToRemove}' removed from inventory.");

break;

}

}

if (!itemFound)

{

Console.WriteLine($"Item '{itemToRemove}' not found in inventory.");

}

}

static void SearchItem()

{

Console.Write("Enter the name of the item to search for: ");

string itemToSearch = Console.ReadLine();

bool itemFound = false;

for (int i = 0; i < itemCount; i++)

{

if (inventory[i] == itemToSearch)

{

Console.WriteLine($"Item '{itemToSearch}' found in inventory.");

itemFound = true;

break;

}

}

if (!itemFound)

{

Console.WriteLine($"Item '{itemToSearch}' not found in inventory.");

}

}

static void DisplayInventory()

{

if (itemCount > 0)

{

Console.WriteLine("Inventory:");

for (int i = 0; i < itemCount; i++)

{

Console.WriteLine($"{i + 1}: {inventory[i]}");

}

}

else

{

Console.WriteLine("Inventory is empty.");

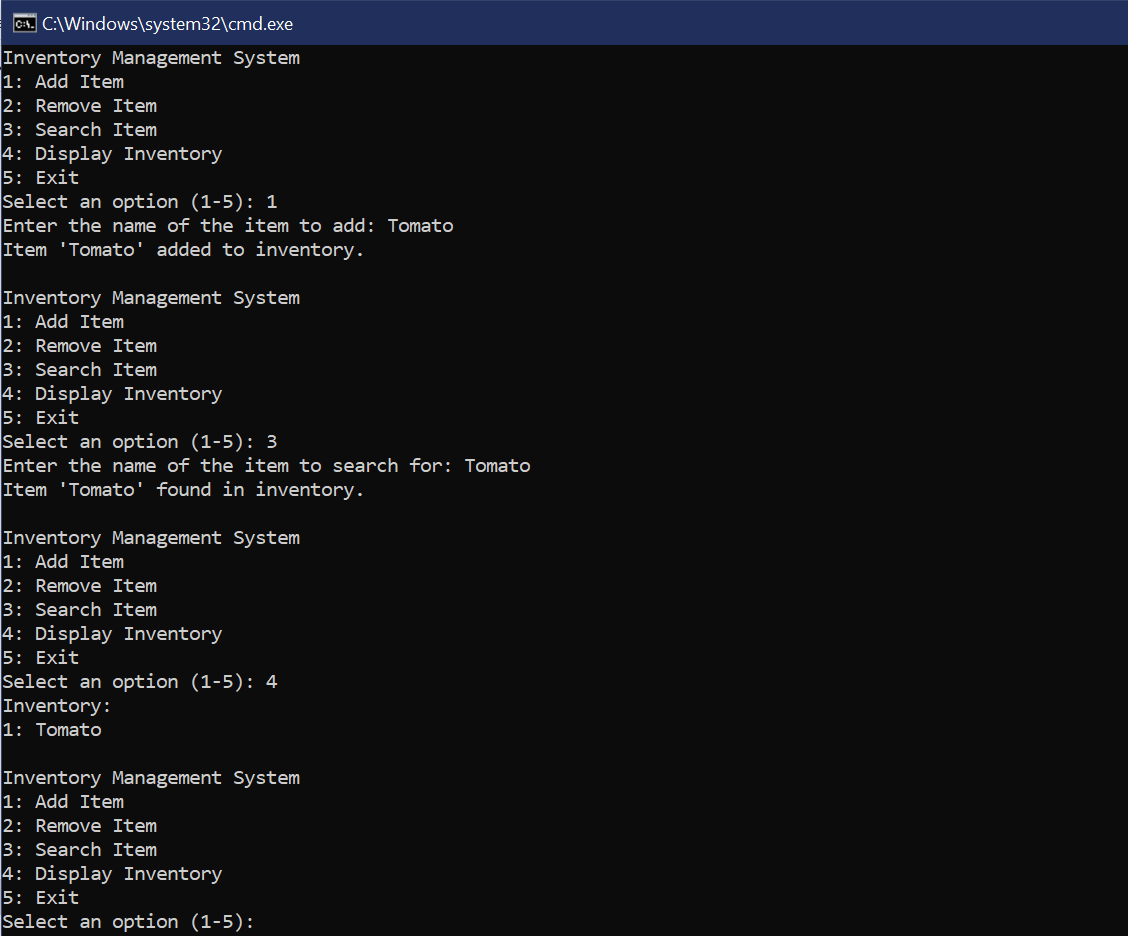
}

}

}

}

Output:



1D Arrays

Task No 03:

Write a program which input 2 matrix of user defined rows and columns and perform following operation:

a. Display/Print as a Matrix

b. Addition of Matrix

c. Subtraction of Matrix

d. matrix multiplication

e. Determinant

f. Inverse

Solution:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab01Task03

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Enter the number of rows for Matrix A");

int rowsA = int.Parse(Console.ReadLine());

Console.WriteLine("Enter the number of Column for Matrix A");

int colA = int.Parse(Console.ReadLine());

Console.WriteLine("Enter the number of rows for Matrix B");

int rowsB = int.Parse(Console.ReadLine());

Console.WriteLine("Enter the number of Column for Matrix B");

int colB = int.Parse(Console.ReadLine());

int[,] arrayA = new int[rowsA, colA];

int[,] arrayB = new int[rowsB, colB];

int rows = arrayA.GetLength(0);

int cols = arrayA.GetLength(1);

int[,] result = new int[rows, cols];

for (int i = 0; i < rowsA; i++)

{

for (int j = 0; j < colA; j++)

{

Console.WriteLine("Enter the Element for Matrix A : \t");

arrayA[i, j] = int.Parse(Console.ReadLine());

}

}

for (int i = 0; i < rowsB; i++)

{

for (int j = 0; j < colB; j++)

{

Console.WriteLine("Enter the Element for Matrix B: \t");

arrayB[i, j] = int.Parse(Console.ReadLine());

}

}

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*Display of Matrix\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("Matrix A is: ");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(arrayA[i, j] + "\t");

}

Console.WriteLine();

}

Console.WriteLine("Matrix B is: ");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(arrayB[i, j] + "\t");

}

Console.WriteLine();

}

int opt;

do

{

Console.WriteLine("What do you want to do with matrices: \n1. Add matrices \n2. Subtract matrices \n3. Multiply matrices \n4. Calculate determinant of Matrix A \n5. Calculate inverse of Matrix A \n6. Exit");

opt = int.Parse(Console.ReadLine());

if (opt == 1)

{

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

result[i, j] = arrayA[i, j] + arrayB[i, j];

}

}

Console.WriteLine("\nAdded Successfully! The answer is:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(result[i, j] + "\t");

}

Console.WriteLine();

}

}

if (opt == 2)

{

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

result[i, j] = arrayA[i, j] - arrayB[i, j];

}

}

Console.WriteLine("\nSubtracted Successfully! The answer is: \n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(result[i, j] + "\t");

}

Console.WriteLine();

}

}

if (opt == 3)

{

if (colA != rowsB)

{

Console.WriteLine("Matrix multiplication is not possible due to incompatible dimensions.");

}

else

{

for (int i = 0; i < rowsA; i++)

{

for (int j = 0; j < colB; j++)

{

result[i, j] = 0;

for (int k = 0; k < colA; k++)

{

result[i, j] += arrayA[i, k] \* arrayB[k, j];

}

}

}

Console.WriteLine("\nMultiplied Successfully! The result is:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(result[i, j] + "\t");

}

Console.WriteLine();

}

}

}

if (opt == 4)

{

{

double determinantA = CalculateDeterminant(arrayA, rowsA);

Console.WriteLine($"Determinant of Matrix A is: {determinantA}");

}

Console.WriteLine("\nDeterminant complete Successfully! The result is:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(result[i, j] + "\t");

}

Console.WriteLine();

}

}

if (opt == 5)

{

double[,] inverseA = CalculateInverse(arrayA, rowsA);

Console.WriteLine("Inverse of Matrix A is:");

for (int i = 0; i < rowsA; i++)

{

for (int j = 0; j < rowsA; j++)

{

Console.Write(inverseA[i, j] + "\t");

}

}

Console.WriteLine("\nInverse Successfully! The result is:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

Console.Write(result[i, j] + "\t");

}

Console.WriteLine();

}

}

} while (opt != 6);

}

static double CalculateDeterminant(int[,] matrix, int n)

{

return 0;

}

static double[,] CalculateInverse(int[,] matrix, int n)

{

double[,] inverse = new double[n, n];

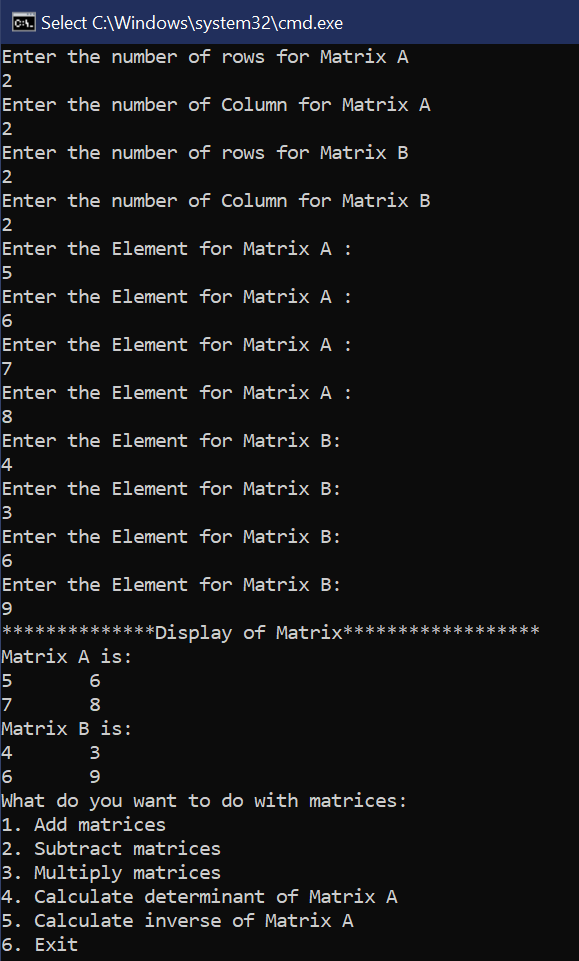
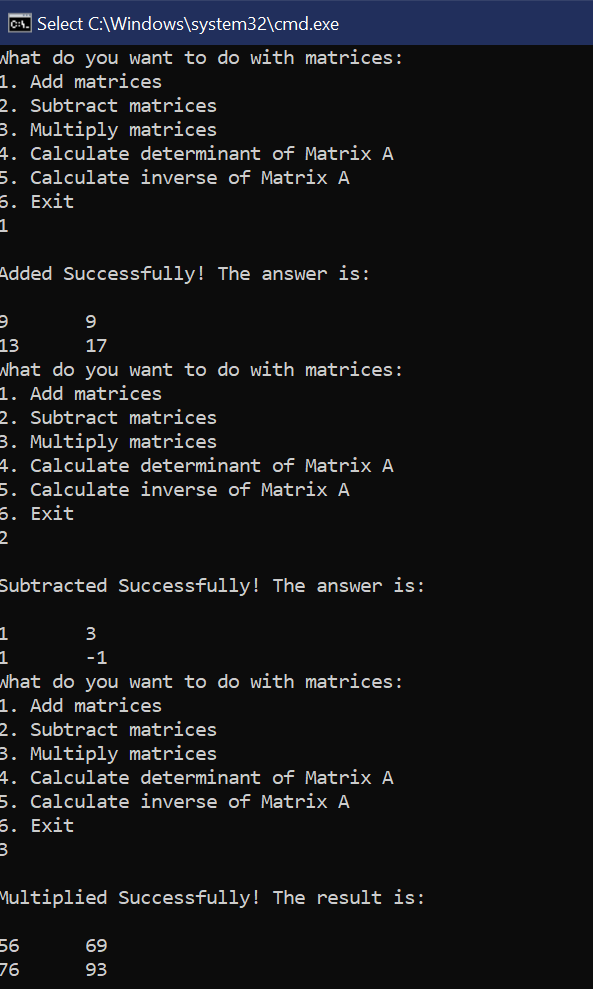
return inverse;

}

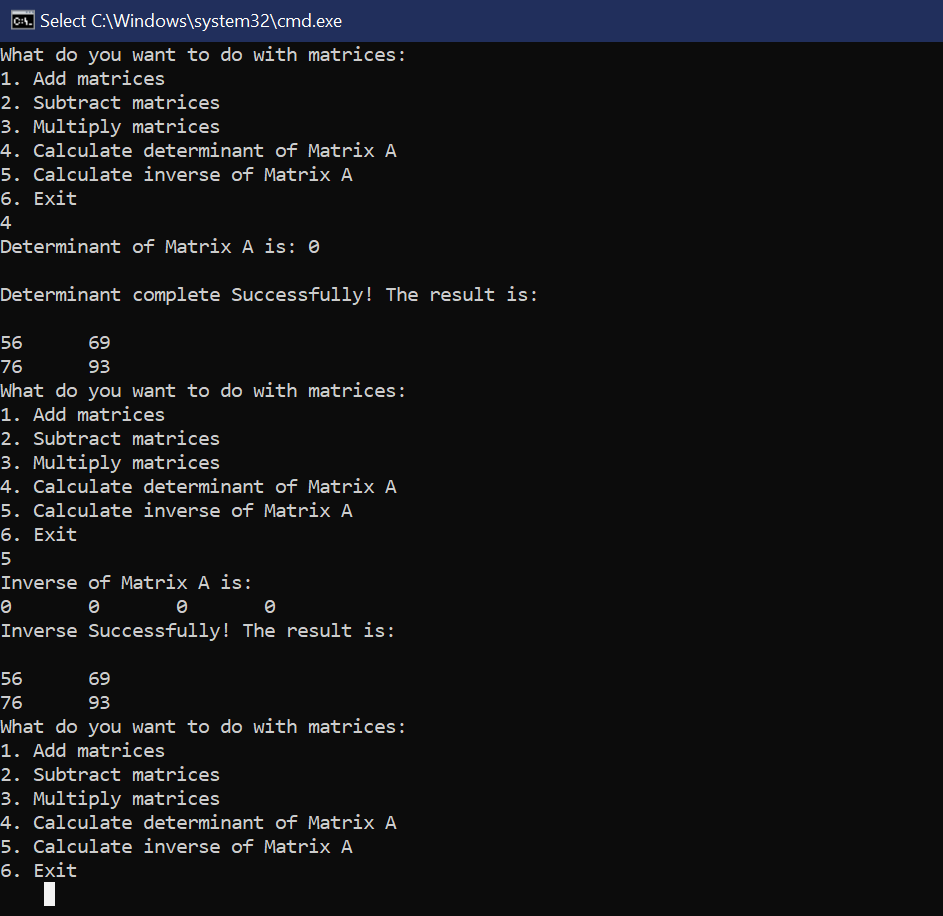
}

}

Output:

. 

*Figure 1 Figure 2*



*Figure 3*