# Best Programming Practice

1. All values as variables including Fixed, User Inputs, and Results
2. Avoid Hard Coding of variables wherever possible
3. Proper naming conventions for all variables
4. Proper Program Name and Class Name
5. Follow proper indentation
6. Give comments for every step or logical block like a variable declaration or conditional and loop blocks
7. Validate user inputs and handle errors gracefully using Console.Error and Environment.Exit.
8. Use Array ***length*** property while using ***for*** loop
9. **Sample Program 1 -** Create a program to find the sum of all the digits of a number given by a user using an array and display the sum.

**Hint =>**

1. Take the input for a number and validate, if failed state and exit the program
2. Find the count of digits in the number
3. Find the digits in the number and save them in an array
4. Find the sum of the digits of the number and display the sum

// Create SumOfDigit Class to compute the sum of all digits of a number using

// an array

using System;

class SumOfDigits

{

static void Main(string[] args)

{

// Take input for a number

Console.Write("Enter a number: ");

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

// Validate the input

if (number < 0)

{

Console.Error.WriteLine("Invalid Number.");

Environment.Exit(0);

}

// Find the count of digits

int temp = number, count = 0;

while (temp > 0)

{

count++;

temp /= 10;

}

// Find the digits and store them in an array

int[] digits = new int[count];

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

{

digits[i] = number % 10;

number /= 10;

}

// Calculate the sum of the digits

int sum = 0;

foreach (int digit in digits)

{

sum += digit;

}

// Display the sum

Console.WriteLine($"Sum of Digits: {sum}");

}

}



1. **Sample Program 2 -** Working with Multi-Dimensional Arrays. Write a C# program to create a 2 Dimensional (2D) array (matrix) of integers, initialize it with values, and print the sum of all elements in the matrix

**Hint =>**

1. Take the input for a number of rows and columns
2. Create a 2D array (matrix) of integers
3. Take the input for the elements of the matrix
4. Calculate the sum of all elements in the matrix and display the sum
5. Also, Display the matrix

using System;

class TwoDArray

{

static void Main(string[] args)

{

// Declare and initialize the 2D Array

int[,] matrix = new int[3, 3];

// Input the elements of the 2D Array

Console.WriteLine("Enter the elements of the 2D Array:");

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

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

}

}

// Display the elements and calculate the sum

int sum = 0;

Console.WriteLine("The elements of the 2D Array are:");

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

Console.Write(matrix[i, j] + " ");

sum += matrix[i, j];

}

Console.WriteLine();

}

// Display the sum

Console.WriteLine($"The sum of the elements of the 2D Array is: {sum}");

}

}



# Level 1 Practice Programs

1. Write a program to take user input for the age of all 10 students in a class and check whether the student can vote depending on his/her age is greater or equal to 18.

**Hint =>**

1. Define an array of 10 integer elements and take user input for the student's age.
2. Loop through the array using the length property and for the element of the array check If the age is a negative number print an invalid age and if 18 or above, print The student with the age \_\_\_ can vote. Otherwise, print The student with the age \_\_\_ cannot vote.

| using System;  class Solution {  public static void Main() {  // Initialize an array and take user input for ages  int[] ages = new int[10];   for (int i = 0; i < ages.Length; i++) {  Console.Write("Age of student {0}: ", i + 1);  ages[i] = int.Parse(Console.ReadLine());  }   // Check voting eligibility based on age  foreach (int age in ages) {  if (age < 0) {  Console.WriteLine("Invalid age");  } else if (age >= 18) {  Console.WriteLine("The student with the age {0} can vote", age);  } else {  Console.WriteLine("The student with the age {0} cannot vote", age);  }  }  } } |
| --- |

1. Write a program to take user input for 5 numbers and check whether a number is positive, negative, or zero. Further for positive numbers check if the number is even or odd. Finally compare the first and last elements of the array and display if they equal, greater or less

**Hint =>**

1. Define an integer array of 5 elements and get user input to store in the array.
2. Loop through the array using the length If the number is positive, check for even or odd numbers and print accordingly
3. If the number is negative, print negative. Else if the number is zero, print zero.
4. Finally compare the first and last element of the array and display if they equal, greater or less

| using System;  class Solution {  public static void Main() {  // Initialize an array and take user input for numbers  int[] numbers = new int[5];   for (int i = 0; i < numbers.Length; i++) {  Console.Write("Number {0}: ", i + 1);  numbers[i] = int.Parse(Console.ReadLine());  }   // Check if the number is positive, negative, or zero and if it's even or odd  foreach (int num in numbers) {  if (num > 0) {  Console.WriteLine("{0} is positive and {1}", num, (num % 2 == 0 ? "even" : "odd"));  } else if (num < 0) {  Console.WriteLine("{0} is negative", num);  } else {  Console.WriteLine("{0} is zero", num);  }  }   // Compare the first and last elements of the array  if (numbers[0] == numbers[numbers.Length - 1]) {  Console.WriteLine("First and last elements are equal.");  } else if (numbers[0] > numbers[numbers.Length - 1]) {  Console.WriteLine("First element is greater than the last element.");  } else {  Console.WriteLine("First element is less than the last element.");  }  } } |
| --- |

1. Create a program to print a multiplication table of a number.

**Hint =>**

1. Get an integer input and store it in the number variable. Also, define a integer array to store the results of multiplication from 1 to 10
2. Run a loop from 1 to 10 and store the results in the multiplication table array
3. Finally, display the result from the array in the format number \* i = \_\_\_

| using System;  class Solution {  public static void Main() {  // Prompt the user to enter a number   Console.Write("Enter a number: ");  int number = int.Parse(Console.ReadLine());   // Initialize an array   int[] table = new int[10];   for (int i = 1; i <= 10; i++) {  table[i - 1] = number \* i;  }   // Display the multiplication table  for (int i = 1; i <= 10; i++) {  Console.WriteLine("{0} \* {1} = {2}", number, i, table[i - 1]);  }  } } |
| --- |

1. Write a program to store multiple values in an array up to a maximum of 10 or until the user enters a 0 or a negative number. Show all the numbers as well as the sum of all numbers

**Hint =>**

1. Create a variable to store an array of 10 elements of type double as well as a variable to store the total of type double initializes to 0.0. Also, the index variable is initialized to 0 for the array
2. Use infinite while loop as in ***while (true)***
3. Take the user entry and check if the user entered 0 or a negative number to break the loop
4. Also, ***break*** from the loop if the index has a value of 10 as the array size is limited to 10.
5. If the user entered a number other than 0 or a negative number inside the while loop then assign the number to the array element and increment the index value
6. Take another ***for*** loop to get the values of each element and add it to the total
7. Finally display the total value

| using System;  class Solution {  public static void Main() {  // Initialize an array and variables for total and index  double[] numbers = new double[10];  double total = 0.0;  int index = 0;   // Take input until 0, a negative number, or 10 inputs  while (true) {  Console.Write("Number {0}: ", index + 1);  double input = double.Parse(Console.ReadLine());   if (input <= 0 || index == 10) {  break;  }   numbers[index++] = input;  }   // Calculate the total of the entered numbers  for (int i = 0; i <= index; i++) {  total += numbers[i];  }   // Display entered numbers and their total  Console.WriteLine("Numbers entered: {0}", string.Join(", ", numbers));  Console.WriteLine("Total: {0}", total);  } } |
| --- |

1. Create a program to find the multiplication table of a number entered by the user from 6 to 9 and display the result

**Hint =>**

1. Take integer input and store it in the variable number as well as define an integer array to store the multiplication result in the variable multiplicationResult
2. Using a for loop, find the multiplication table of numbers from 6 to 9 and save the result in the array
3. Finally, display the result from the array in the format number \* i = \_\_\_

| using System;  class Solution {  public static void Main() {  // prompt for user to enter number  Console.WriteLine("Enter a number : ");  int number = Convert.ToInt(Console.ReadLine());    // Initialize array   int[] table = new int[10];    for (int i = 0; i <= 9; i++) {  table[i] = i \* j;  }  }   // Display the multiplication tables  Console.WriteLine("Multiplication Table for {0}:", number);  for (int i = 1; i <= 10; i++) {  Console.WriteLine("{0} \* {1} = {2}", number, i, table[i-1]);  }  }  } } |
| --- |

1. Create a program to find the mean height of players present in a football team.

**Hint =>**

1. The formula to calculate the mean is: mean = sum of all elements / number of elements
2. Create a double array named heights of size 11 and get input values from the user.
3. Find the sum of all the elements present in the array.
4. Divide the sum by 11 to find the mean height and print the mean height of the football team

| using System;  class Solution {  public static void Main() {  // Initialize an array   double[] heights = new double[11];   // Take input for the heights  for (int i = 0; i < heights.Length; i++) {  Console.Write("Height of player {0}: ", i + 1);  heights[i] = double.Parse(Console.ReadLine());  }   // Calculate the mean height  double total = 0;  foreach (double height in heights) {  total += height;  }  double mean = total / heights.Length;   // Display the height  Console.WriteLine("The mean height of the football team is {0}", mean);  } } |
| --- |

1. Create a program to save odd and even numbers into odd and even arrays between 1 to the number entered by the user. Finally, print the odd and even numbers array

**Hint =>**

1. Get an integer input from the user, assign it to a variable ***number,*** and check for Natural Number. If not a natural number then print an error and exit the program
2. Create an integer array for even and odd numbers with size = number / 2 + 1
3. Create index variables for odd and even numbers and initialize them to zero
4. Using a for loop, iterate from 1 to the number, and in each iteration of the loop, save the odd or even number into the corresponding array
5. Finally, print the odd and even numbers array using the odd and even index

| using System;  class Solution {  public static void Main() {  // Prompt the user to enter a natural number  Console.Write("Enter a natural number: ");  int number = int.Parse(Console.ReadLine());   if (number <= 0) {  Console.WriteLine("Not a natural number.");  }   // Initialize arrays to store odd and even numbers  int[] oddNumbers = new int[number / 2 + 1];  int[] evenNumbers = new int[number / 2 + 1];  int oddIndex = 0, evenIndex = 0;   // Categorize numbers as odd or even  for (int i = 1; i <= number; i++) {  if (i % 2 == 0) {  evenNumbers[evenIndex++] = i;  } else {  oddNumbers[oddIndex++] = i;  }  }   // Display the odd and even numbers  Console.WriteLine("Odd numbers: {0}", string.Join(", ", oddNumbers));  Console.WriteLine("Even numbers: {0}", string.Join(", ", evenNumbers));  } } |
| --- |

1. Create a program to find the factors of a number taken as user input, store the factors in an array, and display the factors

**Hint =>**

1. Take the input for a number
2. Find the factors of the number and save them in an array. For this create integer variable maxFactor and initialize to 10, factors array of size maxFactor and index variable to reflect the index of the array.
3. To find factors loop through the numbers from 1 to the number, find the factors, and add them to the array element by incrementing the index. If the index is equal to maxIndex, then need factors array to store more elements
4. To store more elements, reset the maxIndex to twice its size, use the temp array to store the elements from the factors array, and eventually assign the factors array to the temp array
5. Finally, Display the factors of the number

| using System;  class Solution {  public static void Main() {  // Prompt the user to enter a number  Console.Write("Enter a number : ");  int number = int.Parse(Console.ReadLine());   // Initialize an array to store factors and its size  int maxFactor = 10;  int[] factors = new int[maxFactor];  int index = 0;   // Find factors of the number  for (int i = 1; i <= number; i++) {  if (number % i == 0) {  if (index == maxFactor) {  maxFactor \*= 2;  int[] temp = new int[maxFactor];  Array.Copy(factors, temp, factors.Length);  factors = temp;  }  factors[index++] = i;  }  }   // Display the factors  Console.WriteLine("Factors of {0}: {1}", number, string.Join(", ", factors));  }  } |
| --- |

1. Working with Multi-Dimensional Arrays. Write a C# program to create a 2D Array and Copy the 2D Array into a single dimension array

**Hint =>**

1. Take user input for rows and columns, create a 2D array (Matrix), and take the user input
2. Copy the elements of the matrix to a 1D array. For this create a 1D array of size rows\*columns as in int[] array = new int[rows \* columns];
3. Define the index variable and Loop through the 2D array. Copy every element of the 2D array into the 1D array and increment the index
4. Note: For looping through the 2D array, you will need Nested for loop, Outer for loop for rows, and the inner for loops to access each element

| using System;  class Solution {  public static void Main() {  // Prompt user for dimensions of the 2D array  Console.Write("Enter the number of rows: ");  int rows = int.Parse(Console.ReadLine());    Console.Write("Enter the number of columns: ");  int cols = int.Parse(Console.ReadLine());   // Initialize the 2D array  int[,] twoDArray = new int[rows, cols];   for (int i = 0; i < rows; i++) {  for (int j = 0; j < cols; j++) {  Console.Write("Element at ({0}, {1}): ", i + 1, j + 1 );  twoDArray[i, j] = int.Parse(Console.ReadLine());  }  }   // Convert the 2D array into a 1D array  int[] oneDArray = new int[rows \* cols];  for (int i = 0; i < rows; i++) {  for (int j = 0; j < cols; j++) {  oneDArray[i \* cols + j] = twoDArray[i, j];   }  }   // Display the 1D array  Console.WriteLine("Converted 1D Array: {0}", string.Join(", ", oneDArray));  } } |
| --- |

1. Write a program FizzBuzz, take a number as user input and if it is a positive integer loop from 0 to the number and save the number, but for multiples of 3 save "Fizz" instead of the number, for multiples of 5 save "Buzz", and for multiples of both save "FizzBuzz". Finally, print the array results for each index position in the format Position 1 = 1, …, Position 3 = Fizz,...

**Hint =>**

1. Create a String Array to save the results and
2. Finally, loop again to show the results of the array based on the index position

| using System;  class Solution {  public static void Main() {  // Prompt user to enter a number  Console.WriteLine("Enter a number:");  int number = int.Parse(Console.ReadLine());   // Validate input  if (number <= 0) {  Console.WriteLine("Please enter a positive integer.");  }   // Initialize an array results  string[] results = new string[number + 1];   // Populate the results array   for (int i = 0; i <= number; i++) {  if (i % 3 == 0 && i % 5 == 0) {  results[i] = "FizzBuzz";  }  else if (i % 3 == 0) {  results[i] = "Fizz";  }  else if (i % 5 == 0) {  results[i] = "Buzz";  }  else {  results[i] = i.ToString();  }  }   // Display the results  for (int i = 0; i <= number; i++) {  Console.WriteLine("Position {0} = {1}", i, results[i]);  }  } } |
| --- |