# **Methods**

Program to understand method :

Example No 1:  
  
using System;

namespace MethodExercises

{

internal class Program

{

static void Main(string[] args)

{

Console.WriteLine("Choose an exercise (1, 2, 3, or 4):");

Console.WriteLine("1: Multiplication Table");

Console.WriteLine("2: Square and Cube");

Console.WriteLine("3: Array Conversion and Sum");

Console.WriteLine("4: Reverse Words in a String");

string exercise = Console.ReadLine();

switch (exercise)

{

case "1":

Console.Write("Enter the number for the multiplication table: ");

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

Console.Write("Enter how many times to multiply: ");

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

MultiplicationTable(number, amount);

break;

case "2":

Console.Write("Enter the number to find the square and cube: ");

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

SquareAndCube(nr);

break;

case "3":

Console.Write("Enter numbers separated by a character: ");

string numbers = Console.ReadLine();

Console.Write("Enter the character used as a separator: ");

char separator = Convert.ToChar(Console.ReadLine());

ArrayConverterAndSum(numbers, separator);

break;

case "4":

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

string sentence = Console.ReadLine();

ReverseWords(sentence);

break;

default:

Console.WriteLine("Invalid choice. Please enter 1, 2, 3, or 4.");

break;

}

}

static int[] Conversion(string input, char separator)

{

string[] parts = input.Split(separator);

int[] result = new int[parts.Length];

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

{

result[i] = int.Parse(parts[i]);

}

return result;

}

static void Sum(int[] nums)

{

int sum = 0;

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

{

sum += nums[i];

}

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

}

static void ArrayConverterAndSum(string input, char separator)

{

int[] array = Conversion(input, separator);

Sum(array);

}

static void SquareAndCube(int number)

{

int squared = Square(number);

int cubed = Cube(number);

Console.WriteLine($"The square of {number} is: {squared}");

Console.WriteLine($"The cube of {number} is: {cubed}");

}

static int Cube(int number)

{

return number \* number \* number;

}

static int Square(int number)

{

return number \* number;

}

static void MultiplicationTable(int number, int amount)

{

Console.WriteLine($"Multiplication table for {number}:");

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

{

int result = number \* i;

Console.WriteLine($"{i} \* {number} = {result}");

}

}

static void ReverseWords(string sentence)

{

string[] words = sentence.Split(' ');

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

{

words[i] = ReverseWord(words[i]);

}

string reversedSentence = string.Join(" ", words);

Console.WriteLine($"Reversed words: {reversedSentence}");

}

static string ReverseWord(string word)

{

char[] charArray = word.ToCharArray();

Array.Reverse(charArray);

return new string(charArray);

}

}

}

Example No : 2  
  
using System;

namespace MethodsProject

{

internal class Program

{

static void Main(string[] args)

{

HelloWorld();

int addResult = Add(5, 10);

Console.WriteLine($"Add(5, 10) = {addResult}");

int[] nrs1 = { 4, 5, 1, 0, 2, 3 };

Console.WriteLine("Original nrs1 array:");

Print(nrs1);

Array.Sort(nrs1);

Console.WriteLine("Sorted nrs1 array:");

Print(nrs1);

int[] nrs2 = { 2, 4, 6, 8, 10, 12, 14, 16, 18 };

Console.WriteLine("Original nrs2 array:");

Print(nrs2);

Array.Sort(nrs2);

Console.WriteLine("Sorted nrs2 array:");

Print(nrs2);

int[] nrs3 = Fill();

Console.WriteLine("Generated nrs3 array:");

Print(nrs3);

Array.Sort(nrs3);

Console.WriteLine("Sorted nrs3 array:");

Print(nrs3);

int[] nrs4 = Fill(20);

Console.WriteLine("Generated nrs4 array:");

Print(nrs4);

Array.Sort(nrs4);

Console.WriteLine("Sorted nrs4 array:");

Print(nrs4);

string[] words = { "banana", "apple", "lemon" };

Console.WriteLine("Original words array:");

Print(words);

Array.Sort(words);

Console.WriteLine("Sorted words array:");

Print(words);

double[] doubles = { 1.2, 1.0, 1.01 };

Console.WriteLine("Original doubles array:");

Print(doubles);

Array.Sort(doubles);

Console.WriteLine("Sorted doubles array:");

Print(doubles);

int studentGrade = 75;

Console.WriteLine($"Student's grade BEFORE extra credit: {studentGrade}");

GiveExtraCredit(ref studentGrade);

Console.WriteLine($"Student's grade AFTER extra credit: {studentGrade}");

int[] grades = new int[1] { 80 };

Console.WriteLine($"Student's grade ARRAY BEFORE extra credit: {grades[0]}");

GiveExtraCreditArray(grades);

Console.WriteLine($"Student's grade ARRAY AFTER extra credit: {grades[0]}");

int a = 6;

int b = 8;

Console.WriteLine($"Before swap - a: {a}, b: {b}");

SwapValue(ref a, ref b);

Console.WriteLine($"After swap - a: {a}, b: {b}");

int x = 6;

int y = 8;

Console.WriteLine($"Before swap - x: {x}, y: {y}");

SwapValue1( x, y);

Console.WriteLine($"After swap - x: {x}, y: {y}");

Console.Write("Enter array size: ");

string input = Console.ReadLine();

int[] array;

if (int.TryParse(input, out int size))

{

array = Fill(size);

}

else

{

array = Fill();

}

Print(array);

AddAndMultiply(5, 10, out int sum, out int product);

Console.WriteLine($"Sum: {sum}, Product: {product}");

// method with optional parameters

int result = AddValues(5, z: 7);

Console.WriteLine($"AddValues(5, z: 7) + 5 = {result + 5}");

// params keyword

PrintArray(5, 6, 7, 8, 9, 10, 11, 12, 13, 14);

}

static void HelloWorld()

{

Console.WriteLine("Hello, World!");

}

static int Add(int number1, int number2)

{

return number1 + number2;

}

static void Print(int[] numbers)

{

foreach (var number in numbers)

{

Console.Write(number + " ");

}

Console.WriteLine();

}

static void Print(double[] numbers)

{

foreach (var number in numbers)

{

Console.Write(number + " ");

}

Console.WriteLine();

}

static void Print(string[] words)

{

foreach (var word in words)

{

Console.Write(word + " ");

}

Console.WriteLine();

}

static void GiveExtraCredit(ref int grade)

{

grade += 3;

}

static void GiveExtraCreditArray(int[] grades)

{

grades[0] += 3;

}

static void SwapValue(ref int a, ref int b)

{

int temp = a;

a = b;

b = temp;

}

static void SwapValue1(int a, int b)

{

int temp = a;

a = b;

b = temp;

}

static int[] Fill(int size = 10)

{

Random random = new Random();

int[] array = new int[size];

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

{

array[i] = random.Next(-20, 21);

}

return array;

}

static void AddAndMultiply(int a, int b, out int added, out int multiplied)

{

added = a + b;

multiplied = a \* b;

}

static int AddValues(int x, int y = 1, int z = 4)

{

return x + y + z;

}

static void PrintArray(params int[] array)

{

foreach (var item in array)

{

Console.WriteLine(item);

}

}

}

}

ref Parameters:

* Must be initialized before passing to the method.
* Method can read and modify the parameter.
* Useful when you need to update the caller's variable.

out Parameters:

* Do not need to be initialized before passing.
* Must be assigned within the method.
* Useful for methods that need to return multiple values.