Main

// Подключение пространства имен System для использования встроенных классов, таких как Console.

using System;

// Определение пространства имен проекта, чтобы сгруппировать классы и избежать конфликтов имен.

namespace project01

{

// Определение статического класса Main1, который будет содержать точку входа программы.

public static class Main1

{

// Точка входа программы, метод Main.

static void Main()

{

// Вызов метода Point1A из класса TaskNum1 для выполнения подзадачи 1A.

TaskNum1.Point1A();

// Вызов метода Point1B из класса TaskNum1 для выполнения подзадачи 1B.

TaskNum1.Point1B();

// Вызов метода Point1C из класса TaskNum1 для выполнения подзадачи 1C.

TaskNum1.Point1C();

// Вызов метода Point1D из класса TaskNum1 для выполнения подзадачи 1D.

TaskNum1.Point1D();

// Вызов метода Point1E из класса TaskNum1 для выполнения подзадачи 1E.

TaskNum1.Point1E();

// Вызов метода Point1F из класса TaskNum1 для выполнения подзадачи 1F.

TaskNum1.Point1F();

// Ожидание ввода пользователя перед продолжением, чтобы дать время просмотреть результаты.

Console.ReadLine();

// Вызов метода Point2A из класса TaskNum2 для выполнения подзадачи 2A.

TaskNum2.Point2A();

// Вызов метода Point2B из класса TaskNum2 для выполнения подзадачи 2B.

TaskNum2.Point2B();

// Вызов метода Point2C из класса TaskNum2 для выполнения подзадачи 2C.

TaskNum2.Point2C();

// Вызов метода Point2D из класса TaskNum2 для выполнения подзадачи 2D.

TaskNum2.Point2D();

// Ожидание ввода пользователя перед продолжением к следующему блоку задач.

Console.ReadLine();

// Вызов метода Point3A из класса TaskNum3 для выполнения подзадачи 3A.

TaskNum3.Point3A();

// Вызов метода Point3B из класса TaskNum3 для выполнения подзадачи 3B.

TaskNum3.Point3B();

// Вызов метода Point3C из класса TaskNum3 для выполнения подзадачи 3C.

TaskNum3.Point3C();

// Ожидание ввода пользователя перед переходом к заключительным задачам.

Console.ReadLine();

// Вызов метода Point4 из класса TaskNum4 для выполнения задачи 4.

TaskNum4.Point4();

// Вызов метода Point5 из класса TaskNum5 для выполнения задачи 5.

TaskNum5.Point5();

// Вызов метода Point6 из класса TaskNum6 для выполнения задачи 6.

TaskNum6.Point6();

}

}

}

TaskNum1

// Подключение пространства имен System для использования встроенных классов, таких как Console.

using System;

// Определение статического класса TaskNum1, содержащего методы для выполнения различных подзадач.

public static class TaskNum1

{

// Метод Point1A выполняет ввод и вывод значений различных типов данных.

public static void Point1A()

{

Console.WriteLine("-------------Task 1.a-------------"); // Заголовок для задачи 1a.

// Ввод и вывод для sbyte

Console.WriteLine("sbyte: ");

sbyte sbyteType = Convert.ToSByte(Console.ReadLine()); // Чтение и преобразование в sbyte.

// Ввод и вывод для byte

Console.WriteLine("byte: ");

byte byteType = Convert.ToByte(Console.ReadLine()); // Чтение и преобразование в byte.

// Ввод и вывод для short

Console.WriteLine("short: ");

short shortType = Convert.ToInt16(Console.ReadLine()); // Чтение и преобразование в short.

// Ввод и вывод для ushort

Console.WriteLine("ushort: ");

ushort ushortType = Convert.ToUInt16(Console.ReadLine()); // Чтение и преобразование в ushort.

// Ввод и вывод для int

Console.WriteLine("int: ");

int intType = Convert.ToInt32(Console.ReadLine()); // Чтение и преобразование в int.

// Ввод и вывод для uint

Console.WriteLine("uint: ");

uint uintType = Convert.ToUInt32(Console.ReadLine()); // Чтение и преобразование в uint.

// Ввод и вывод для long

Console.WriteLine("long: ");

long longType = Convert.ToInt64(Console.ReadLine()); // Чтение и преобразование в long.

// Ввод и вывод для ulong

Console.WriteLine("ulong: ");

ulong ulongType = Convert.ToUInt64(Console.ReadLine()); // Чтение и преобразование в ulong.

// Ввод и вывод для float

Console.WriteLine("float: ");

float floatType = Convert.ToSingle(Console.ReadLine()); // Чтение и преобразование в float.

// Ввод и вывод для double

Console.WriteLine("double: ");

double doubleType = Convert.ToDouble(Console.ReadLine()); // Чтение и преобразование в double.

// Ввод и вывод для decimal

Console.WriteLine("decimal: ");

decimal decimalType = Convert.ToDecimal(Console.ReadLine()); // Чтение и преобразование в decimal.

// Ввод и вывод для bool

Console.WriteLine("bool: ");

bool boolType = Convert.ToBoolean(Console.ReadLine()); // Чтение и преобразование в bool.

// Ввод и вывод для char

Console.WriteLine("char: ");

char charType = Convert.ToChar(Console.ReadLine() ?? ""); // Чтение и преобразование в char.

// Вывод значений всех введенных переменных.

Console.WriteLine($"sbyte: {sbyteType}");

Console.WriteLine($"byte: {byteType}");

Console.WriteLine($"short: {shortType}");

Console.WriteLine($"ushort: {ushortType}");

Console.WriteLine($"int: {intType}");

Console.WriteLine($"uint: {uintType}");

Console.WriteLine($"long: {longType}");

Console.WriteLine($"ulong: {ulongType}");

Console.WriteLine($"float: {floatType}");

Console.WriteLine($"double: {doubleType}");

Console.WriteLine($"decimal: {decimalType}");

Console.WriteLine($"bool: {boolType}");

Console.WriteLine($"char: {charType}");

}

//------------------------------------------------------------------------------------------------

// Метод Point1B показывает примеры явного и неявного приведения типов данных.

public static void Point1B()

{

Console.WriteLine("\n\n-------------Task 1.b-------------"); // Заголовок для задачи 1b.

Console.WriteLine("явное приведение:\n"); // Уведомление о начале примеров явного приведения.

// Примеры явного приведения

int intB = 10;

double doubleB = (double)intB; // Преобразование int в double.

Console.WriteLine($"{intB} --> {doubleB}");

float floatB = 3.14f;

int intB2 = (int)floatB; // Преобразование float в int (теряется дробная часть).

Console.WriteLine($"{floatB} --> {intB2}");

long longB = 1000;

short shortB = (short)longB; // Преобразование long в short.

Console.WriteLine($"{longB} --> {shortB}");

byte byteB = 65;

char charB = (char)byteB; // Преобразование byte в char.

Console.WriteLine($"{byteB} --> {charB}");

double doubleB2 = 3.99;

int intB3 = (int)doubleB2; // Преобразование double в int (теряется дробная часть).

Console.WriteLine($"{doubleB2} --> {intB3}");

Console.WriteLine("Неявное приведение:\n"); // Уведомление о начале примеров неявного приведения.

// Примеры неявного приведения

int intB4 = 5;

double doubleB3 = intB4; // Неявное преобразование int в double.

Console.WriteLine($"{intB4} --> {doubleB3}");

short shortB2 = 100;

int intB5 = shortB2; // Неявное преобразование short в int.

Console.WriteLine($"{shortB2} --> {intB5}");

float floatB2 = 2.5f;

double doubleB4 = floatB2; // Неявное преобразование float в double.

Console.WriteLine($"{floatB2} --> {doubleB4}");

byte byteB2 = 50;

int intB6 = byteB2; // Неявное преобразование byte в int.

Console.WriteLine($"{byteB2} --> {intB6}");

long longB2 = 1000;

float floatB3 = longB2; // Неявное преобразование long в float.

Console.WriteLine($"{longB2} --> {floatB3}");

}

//------------------------------------------------------------------------------------------------

// Метод Point1C показывает процесс упаковки и распаковки данных.

public static void Point1C()

{

Console.WriteLine("\n\n-------------Task 1.c-------------"); // Заголовок для задачи 1c.

int intC = 42; // Создание переменной типа int.

object boxedC = intC; // Упаковка int в object.

int unpackedC = (int)boxedC; // Распаковка значения обратно в int.

// Вывод оригинального, упакованного и распакованного значения.

Console.WriteLine($"intC: {intC}");

Console.WriteLine($"boxedC: {boxedC}");

Console.WriteLine($"unpackedC: {unpackedC}");

}

//------------------------------------------------------------------------------------------------

// Метод Point1D показывает использование неявной типизации с var.

public static void Point1D()

{

Console.WriteLine("\n\n-------------Task 1.d-------------"); // Заголовок для задачи 1d.

var message = "message"; // Переменная типа string, неявно типизированная.

var number = 5; // Переменная типа int, неявно типизированная.

// Вывод типа и значения переменной message.

Console.WriteLine($"Тип переменной message: {message.GetType()}");

Console.WriteLine($"Значение переменной message: {message}");

// Вывод типа и значения переменной number.

Console.WriteLine($"\nТип переменной number: {number.GetType()}");

Console.WriteLine($"Значение переменной number: {number}");

}

//------------------------------------------------------------------------------------------------

// Метод Point1E показывает работу с nullable-типом.

public static void Point1E()

{

Console.WriteLine("\n\n-------------Task 1.e-------------"); // Заголовок для задачи 1e.

int? nullable = null; // Создание nullable переменной типа int.

Console.WriteLine($"nullable: {nullable}"); // Вывод null-значения.

nullable = 10; // Присвоение значения nullable переменной.

Console.WriteLine($"New nullable: {nullable}"); // Вывод нового значения.

}

//------------------------------------------------------------------------------------------------

// Метод Point1F показывает особенности неявной типизации с var.

public static void Point1F()

{

Console.WriteLine("\n\n-------------Task 1.f-------------"); // Заголовок для задачи 1f.

var varF = 5; // Переменная типа int, неявно типизированная.

Console.WriteLine($"varF: {varF}"); // Вывод значения varF.

// varF = 3.14; // Ошибка: нельзя присвоить значение другого типа переменной, объявленной как var.

// Console.WriteLine($"new varF"); // Выводить это значение не получится.

}

}

TaskNum2

// Подключение пространства имен System для использования стандартных классов, таких как Console.

using System;

// Подключение пространства имен System.Text для использования класса StringBuilder в 2.d.

using System.Text;

// Определение статического класса TaskNum2, который содержит методы для различных задач, связанных со строками.

public static class TaskNum2

{

// Метод Point2A проверяет строки на равенство.

public static void Point2A()

{

Console.WriteLine("\n\n-------------Task 2.a-------------"); // Заголовок задачи 2a.

// Создание строк str1 и str2.

string str1 = "Hello";

string str2 = "World";

// Проверка, равны ли str1 и str2, и вывод результата.

if (str1 == str2)

Console.WriteLine($"{str1} == {str2}");

else

Console.WriteLine($"{str1} != {str2}");

// Создание строки str3, идентичной str1.

string str3 = "Hello";

// Проверка, равны ли str1 и str3, и вывод результата.

if (str1 == str3)

Console.WriteLine($"{str1} == {str3}");

else

Console.WriteLine($"{str1} != {str3}");

}

//------------------------------------------------------------------------------------------------

// Метод Point2B демонстрирует операции со строками: сцепление, копирование, выделение подстроки и другие.

public static void Point2B()

{

Console.WriteLine("\n\n-------------Task 2.b-------------"); // Заголовок задачи 2b.

// Создание строк для операций.

string stringB1 = "String1";

string stringB2 = "StringNum2";

string stringB3 = "lorem ipsum,qwerty";

string stringB4 = "lorem";

// Сцепление строк stringB1 и stringB2.

string concString = stringB1 + stringB2;

Console.WriteLine($"Сцепление: {concString}");

// Копирование строки stringB2.

string copyString = stringB2.Substring(0);

Console.WriteLine($"Копирование: {copyString}");

// Выделение подстроки из stringB3.

string subString = stringB3.Substring(0, 3);

Console.WriteLine($"Выделение подстроки строки {stringB3}: {subString}\n");

// Разделение строки stringB3 на слова по символам пробела и запятой.

string[] stringWords = stringB3.Split(' ', ',');

Console.WriteLine("Разделение строки на слова:");

foreach (string word in stringWords)

{

Console.WriteLine(word); // Вывод каждого слова.

}

// Вставка подстроки subString в stringB3.

string insertString = stringB3.Insert(5, subString);

Console.WriteLine($"\nВставка подстроки {subString} в строку {stringB3}: {insertString}");

// Удаление подстроки из stringB4.

string removeString = stringB4.Remove(3, 2);

Console.WriteLine($"Удаление подстроки в строке {stringB4}: {removeString}");

}

//------------------------------------------------------------------------------------------------

// Метод Point2C проверяет строки на null и пустые значения.

public static void Point2C()

{

Console.WriteLine("\n\n-------------Task 2.c-------------"); // Заголовок задачи 2c.

// Создание строковых переменных, включая пустую строку и null.

string? stringC1 = "";

string? stringC2 = null;

string? stringC3 = "hello";

// Проверка, является ли stringC1 пустой или null.

if (string.IsNullOrEmpty(stringC1))

Console.WriteLine("stringC1 is null or empty");

else

Console.WriteLine("stringC1 is not null or empty");

// Проверка, является ли stringC2 пустой или null.

if (string.IsNullOrEmpty(stringC2))

Console.WriteLine("stringC2 is null or empty");

else

Console.WriteLine("stringC2 is not null or empty");

// Проверка, является ли stringC3 пустой или null.

if (string.IsNullOrEmpty(stringC3))

Console.WriteLine("stringC3 is null or empty");

else

Console.WriteLine("stringC3 is not null or empty");

// Создание строковых переменных, содержащих только пробелы.

string? stringC4 = " ";

string? stringC5 = " \n";

// Проверка, содержит ли stringC4 только пробелы или является null.

if (string.IsNullOrWhiteSpace(stringC4))

Console.WriteLine("\nstringC4 состоит только из пробелов или null");

else

Console.WriteLine("stringC4 состоит не только из пробелов и не null");

// Проверка, содержит ли stringC5 только пробелы или является null.

if (string.IsNullOrWhiteSpace(stringC5))

Console.WriteLine("stringC5 состоит только из пробелов или null");

else

Console.WriteLine("stringC5 состоит не только из пробелов и не null");

// Проверка, содержит ли stringC2 только пробелы или является null.

if (string.IsNullOrWhiteSpace(stringC2))

Console.WriteLine("stringC2 состоит только из пробелов или null");

else

Console.WriteLine("stringC2 состоит не только из пробелов и не null");

}

//------------------------------------------------------------------------------------------------

// Метод Point2D демонстрирует работу с StringBuilder для изменения строк.

public static void Point2D()

{

Console.WriteLine("\n\n-------------Task 2.d-------------"); // Заголовок задачи 2d.

// Создание StringBuilder с исходной строкой.

StringBuilder sb = new StringBuilder("null string");

Console.WriteLine($"Исходная строка: {sb}");

// Удаление подстроки "null" из строки StringBuilder.

sb.Remove(0, 5);

Console.WriteLine($"Удаление подстроки null: {sb}");

// Вставка строки "space " в начало и добавление "!" в конец.

sb.Insert(0, "space ");

sb.Append("!");

Console.WriteLine($"Добавление space в начало и ! в конец: {sb}");

}

}

TaskNum3

// Подключение пространства имен System для использования стандартных классов, таких как Console.

using System;

// Определение статического класса TaskNum3, который содержит методы для различных задач, связанных с массивами.

public static class TaskNum3

{

// Метод Point3A демонстрирует вывод двумерного массива на экран.

public static void Point3A()

{

Console.WriteLine("\n\n-------------Task 3.a-------------"); // Заголовок задачи 3.a.

// Инициализация двумерного массива размером 3x3 и заполнение его значениями.

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

{

{ 2, 34, 5 },

{ 4, 23, 78 },

{ 12, 67, 24}

};

// Цикл для перебора строк массива.

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

{

// Цикл для перебора столбцов массива.

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

{

Console.Write(matrix[i, j] + " "); // Вывод элемента массива и отступ.

}

Console.WriteLine(); // Переход на новую строку после вывода одной строки массива.

}

}

//------------------------------------------------------------------------------------------------

// Метод Point3B демонстрирует работу с одномерным массивом строк, заменяя элемент по индексу.

public static void Point3B()

{

Console.WriteLine("\n\n-------------Task 3.b-------------"); // Заголовок задачи 3.b.

// Инициализация строкового массива.

string[] stringArray = new string[] { "lorem", "ipsum", "hello", "world" };

// Вывод всех элементов массива.

Console.Write("Массив слов: ");

foreach (string str in stringArray)

Console.Write(str + " ");

// Вывод длины массива.

Console.WriteLine($"\nДлина массива: {stringArray.Length}");

// Запрос у пользователя номера слова для замены.

Console.WriteLine("Введите цифру от 1 до 4: ");

int numWord = Convert.ToInt32(Console.ReadLine()) - 1;

// Проверка, находится ли номер в допустимом диапазоне.

if (numWord < 0 || numWord > 4)

return;

// Запрос нового слова от пользователя.

Console.WriteLine("Слово: ");

string word = Console.ReadLine() ?? "";

// Цикл для замены элемента в массиве на новое слово.

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

{

if (numWord == i) // Проверка, соответствует ли индекс выбранному пользователем номеру.

{

stringArray[i] = word; // Замена элемента массива.

break;

}

}

// Вывод нового массива после замены.

Console.Write("Новый массив слов: ");

foreach (string str in stringArray)

Console.Write(str + " ");

}

//------------------------------------------------------------------------------------------------

// Метод Point3C демонстрирует создание и ввод элементов зубчатого массива.

public static void Point3C()

{

Console.WriteLine("\n\n-------------Task 3.c-------------"); // Заголовок задачи 3.c.

// Создание зубчатого массива с тремя подмассивами.

int[][] array = new int[3][];

// Заполнение зубчатого массива, задавая длину каждого подмассива.

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

array[i] = new int[i + 2];

// Запрос ввода элементов массива от пользователя.

Console.WriteLine("Введите элементы массивов массива array: ");

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

{

Console.WriteLine($"Массив {i + 1}: ");

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

{

// Чтение значения для каждого элемента подмассива.

array[i][j] = Convert.ToInt32(Console.ReadLine());

}

}

// Вывод элементов зубчатого массива на экран.

Console.WriteLine("Результат:");

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

{

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

{

Console.Write(array[i][j] + " "); // Вывод элемента массива.

}

Console.WriteLine(); // Переход на новую строку после вывода одного подмассива.

}

}

//------------------------------------------------------------------------------------------------

// Метод Point3D демонстрирует использование var для определения типа переменных и вывод их типов.

public static void Point3D()

{

Console.WriteLine("\n\n-------------Task 3.d-------------"); // Заголовок задачи 3.d.

// Определение массива целых чисел с использованием var.

var array3D = new[] { 1, 2, 3, 4, 5 };

// Определение строки с использованием var.

var string3D = "Hello, World!";

// Вывод типа переменной array3D.

Console.WriteLine("Тип переменной array: " + array3D.GetType());

// Вывод типа переменной string3D.

Console.WriteLine("Тип переменной str: " + string3D.GetType());

}

}

TaskNum456

// Подключение пространства имен System для использования стандартных классов, таких как Console.

using System;

// Определение статического класса TaskNum4, содержащего метод для работы с кортежами.

public static class TaskNum4

{

// Метод Point4 демонстрирует создание и работу с кортежами.

public static void Point4()

{

Console.WriteLine("\n\n-------------Task 4-------------"); // Заголовок задачи 4.

// Создание кортежа с пятью элементами разных типов.

(int, string, char, string, ulong) tuple = (10, "Hello", 'A', "World", 456);

// Вывод всего кортежа.

Console.WriteLine($"Кортеж полностью: {tuple}");

// Выборочный вывод элементов кортежа.

Console.WriteLine($"Кортеж выборочно: {tuple.Item1} {tuple.Item3} {tuple.Item4}");

// Распаковка кортежа в именованные переменные.

(int int1, string string1, char char1, string string2, ulong ulong1) = (10, "Hello", 'A', "World", 456);

Console.WriteLine("\nРаспаковка кортежа в переменные:");

Console.WriteLine($"int1: {int1}");

Console.WriteLine($"string1: {string1}");

Console.WriteLine($"char1: {char1}");

Console.WriteLine($"string2: {string2}");

Console.WriteLine($"ulong1: {ulong1}");

// Создание кортежа с автоматическим определением типов через var.

var tupleVar = (10, "hi", 'D');

Console.WriteLine("\nРаспаковка кортежа без указания типов: ");

// Вывод значений кортежа tupleVar.

Console.WriteLine(tupleVar.Item1);

Console.WriteLine(tupleVar.Item2);

Console.WriteLine(tupleVar.Item3);

// Распаковка отдельных элементов кортежа, игнорируя неиспользуемые значения с помощью "\_".

(\_, string a, \_) = tupleVar;

Console.WriteLine(a); // Вывод только выбранного элемента кортежа.

}

}

//------------------------------------------------------------------------------------------------

// Определение статического класса TaskNum5, содержащего метод для работы с массивами и кортежами.

public static class TaskNum5

{

// Метод Point5 демонстрирует использование кортежа для возврата нескольких значений из метода.

public static void Point5()

{

Console.WriteLine("\n\n-------------Task 5-------------"); // Заголовок задачи 5.

// Инициализация массива целых чисел.

int[] numbers = { 8, 8, 0, 0, 5, 5, 5, 3, 5, 3, 5 };

// Инициализация строки.

string myString = "Hello";

// Определение локальной функции, которая принимает массив и строку, возвращая кортеж.

(int max, int min, int sum, char firstChar) tuple5(int[] arr, string str)

{

int maxNum = arr[0]; // Инициализация переменной для максимального элемента.

int minNum = arr[0]; // Инициализация переменной для минимального элемента.

int sumNum = 0; // Инициализация переменной для суммы элементов.

// Проход по всем элементам массива для нахождения максимального, минимального и суммы.

foreach (int num in arr)

{

if (num > maxNum)

{

maxNum = num;

}

if (num < minNum)

{

minNum = num;

}

sumNum += num;

}

// Получение первого символа строки.

char firstLetter = str[0];

// Возврат кортежа с максимальным, минимальным, суммой элементов массива и первым символом строки.

return (maxNum, minNum, sumNum, firstLetter);

}

// Вызов функции tuple5 и получение кортежа с результатами.

var result = tuple5(numbers, myString);

// Вывод результатов.

Console.WriteLine("Результаты:");

Console.WriteLine($"Максимальный элемент: {result.max}");

Console.WriteLine($"Минимальный элемент: {result.min}");

Console.WriteLine($"Сумма элементов: {result.sum}");

Console.WriteLine($"Первая буква строки: {result.firstChar}\n");

// Инициализация двух кортежей для сравнения.

var tuple5b = (5, "hello", 3.14);

var tuple5c = (1, "world", 3.16);

// Вывод сравнения кортежей.

Console.WriteLine($"Сравнение кортежей {tuple5b} и {tuple5c} ");

// Сравнение кортежей и вывод результата.

if (Tuple.Equals(tuple5b, tuple5c))

{

Console.WriteLine("Кортежи равны");

}

else

{

Console.WriteLine("Кортежи не равны");

}

}

}

//------------------------------------------------------------------------------------------------

// Определение статического класса TaskNum6, содержащего методы для демонстрации checked и unchecked операций.

public static class TaskNum6

{

// Метод Point6 демонстрирует работу с проверенными и непроверенными арифметическими операциями.

public static void Point6()

{

Console.WriteLine("\n\n-------------Task 6-------------"); // Заголовок задачи 6.

// Вызов метода CheckedOp для демонстрации checked операций.

CheckedOp();

// Вызов метода UncheckedOp для демонстрации unchecked операций.

UncheckedOp();

}

// Метод CheckedOp демонстрирует проверку переполнения.

static void CheckedOp()

{

// Вложенная локальная функция для выполнения арифметической операции в checked блоке.

void CheckedOperation()

{

checked

{

int maxValue = int.MaxValue; // Присвоение максимального значения типа int.

maxValue++; // Попытка переполнения значения int.

// Вывод значения после переполнения.

Console.WriteLine($"Максимальное значение с checked: {maxValue}");

}

}

// Попытка выполнения checked операции с обработкой исключения переполнения.

try

{

CheckedOperation();

}

catch(Exception e)

{

// Вывод сообщения об ошибке при переполнении.

Console.WriteLine($"{e.Message}");

}

}

// Метод UncheckedOp демонстрирует операцию без проверки переполнения.

static void UncheckedOp()

{

// Вложенная локальная функция для выполнения арифметической операции в unchecked блоке.

void UncheckedOperation()

{

unchecked

{

// Присвоение значения больше максимального значения int.

int maxValue = int.MaxValue + 1;

// Вывод значения после переполнения в unchecked блоке.

Console.WriteLine($"Максимальное значение с unchecked: {maxValue}");

}

}

// Попытка выполнения unchecked операции с обработкой исключений.

try

{

UncheckedOperation();

}

catch (Exception e)

{

// Вывод сообщения об ошибке при переполнении.

Console.WriteLine($"{e.Message}");

}

}

}