1. **Массивы**

**Объявление и инициализация:**

// Массив целых чисел (фиксированный размер)

int[] numbers = new int[5]; // [0, 0, 0, 0, 0]

int[] primes = { 2, 3, 5, 7, 11 }; // Инициализация значениями

**Доступ к элементам:**

primes[0] = 2; // Индексация с нуля

Console.WriteLine(primes[2]); // Выведет 5

**Перебор массива:**

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

{

Console.WriteLine(primes[i]);

}

// Или через foreach:

foreach (int num in primes)

{

Console.WriteLine(num);

}

**Важно:**

* Размер массива нельзя изменить после создания.
* Выход за границы массива вызывает ошибку (IndexOutOfRangeException).

2. **Списки (**List<T>**)**

**Особенности:**

* Динамический размер.
* Поддерживают добавление, удаление, поиск элементов.

**Использование:**

using System.Collections.Generic; // Не забудьте подключить!

List<string> names = new List<string>(); // Пустой список

names.Add("Анна"); // Добавление элемента

names.Add("Пётр");

names.Remove("Анна"); // Удаление элемента

names[0] = "Мария"; // Изменение элемента

// Инициализация с значениями:

List<int> numbers = new List<int> { 10, 20, 30 };

**Основные методы:**

* Add() — добавляет элемент.
* Remove() — удаляет элемент.
* Contains() — проверяет наличие элемента.
* Count — возвращает количество элементов.

**Пример:**

List<int> scores = new List<int>();

scores.Add(85);

scores.Add(92);

Console.WriteLine(scores.Contains(85)); // True

Console.WriteLine(scores.Count); // 2

3. **Словари (**Dictionary<TKey, TValue>**)**

**Особенности:**

* Хранят пары «ключ-значение».
* Ключи уникальны.

**Использование:**

Dictionary<string, int> ages = new Dictionary<string, int>();

ages["Анна"] = 25; // Добавление элемента

ages.Add("Пётр", 30); // Альтернативный способ

// Проверка наличия ключа:

if (ages.ContainsKey("Анна"))

{

Console.WriteLine(ages["Анна"]); // 25

}

**Перебор словаря:**

foreach (var pair in ages)

{

Console.WriteLine($"{pair.Key}: {pair.Value} лет");

}

4. **Другие коллекции (кратко)**

* **Stack<T>** — «последний вошёл, первый вышел» (LIFO).
* **Queue<T>** — «первый вошёл, первый вышел» (FIFO).
* **HashSet<T>** — хранит уникальные элементы.

**Практическое задание**

**Задача 1: Список покупок**

Создайте программу, которая:

1. Позволяет пользователю добавлять товары в список.
2. Выводит все товары.
3. Удаляет товар по названию.

**Пример:**

List<string> shoppingList = new List<string>();

while (true)

{

Console.WriteLine("1. Добавить | 2. Удалить | 3. Показать | 4. Выход");

string choice = Console.ReadLine();

switch (choice)

{

case "1":

Console.Write("Введите товар: ");

shoppingList.Add(Console.ReadLine());

break;

case "2":

Console.Write("Введите товар для удаления: ");

string item = Console.ReadLine();

if (shoppingList.Contains(item))

shoppingList.Remove(item);

break;

case "3":

foreach (var product in shoppingList)

Console.WriteLine(product);

break;

case "4":

return;

}

}

**Задача 2: Учёт студентов**

Создайте словарь для хранения студентов и их оценок:

Dictionary<string, int> students = new Dictionary<string, int>();

students.Add("Анна", 90);

students.Add("Пётр", 85);

// Вывод всех студентов с оценкой выше 80:

foreach (var student in students)

{

if (student.Value > 80)

Console.WriteLine($"{student.Key}: {student.Value}");

}

**Задача 3: Поиск в массиве**

Напишите функцию, которая ищет число в массиве и возвращает его индекс (или -1, если не найдено):

static int FindIndex(int[] array, int target)

{

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

{

if (array[i] == target)

return i;

}

return -1;

}

// Использование:

int[] numbers = { 5, 10, 15, 20 };

Console.WriteLine(FindIndex(numbers, 15)); // 2