Локальные функции представляют функции, определенные внутри других методов. Локальная функция, как правило, содержит действия, которые применяются только в рамках ее метода.

Например, определим метод который сравнивают сумму чисел двух массивов:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | void Compare(int[] numbers1, int[] numbers2)  {  int numbers1Sum = 0;  int numbers2Sum = 0;    foreach(int number in numbers1)  numbers1Sum += number;    foreach (int number in numbers2)  numbers2Sum += number;    if (numbers1Sum > numbers2Sum)  Console.WriteLine("сумма чисел из массива numbers1 больше");  else if (numbers1Sum < numbers2Sum)  Console.WriteLine("сумма чисел из массива numbers2 больше");  else  Console.WriteLine("суммы чисел обоих массивов равны");  }    int[] numbers1 = { 1, 2, 3 };  int[] numbers2 = { 3, 4, 5, 6, 7 };    Compare(numbers1, numbers2); |

Здесь метод Compare принимает два массива и последовательно вычисляет сумму их элементов, чтобы узнать в каком массиве сумма чисел больше. Несмотря на то, что все работает, здесь есть один недостаток: здесь повторяется действия, которые вычисляют сумму массива:

|  |  |
| --- | --- |
| 1  2  3  4 | int numbers1Sum = 0;    foreach(int number in numbers1)  numbers1Sum += number; |

К тому а что, если мы захотим сравнивать сумму только положительных или четных чисел или как-то иначе изменить логику сравнения? В этом лучше вынести повторяющиеся действия в отдельный метод. Однако если эти действия нигде больше в прогамме не будут вызываться и будут использоваться только в одном методе, то целесообразно определить эти действия в виде локальной функции. Для этого изменим метод Compare следующим образом:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | void Compare(int[] numbers1, int[] numbers2)  {  int numbers1Sum = Sum(numbers1);  int numbers2Sum = Sum(numbers2);    if (numbers1Sum > numbers2Sum)  Console.WriteLine("сумма чисел из массива numbers1 больше");  else if (numbers1Sum < numbers2Sum)  Console.WriteLine("сумма чисел из массива numbers2 больше");  else  Console.WriteLine("суммы чисел обоих массивов равны");    int Sum(int[] numbers)  {  int result = 0;  foreach (int number in numbers)  result += number;  return result;  }  }    int[] numbers1 = { 1, 2, 3 };  int[] numbers2 = { 3, 4, 5, 6, 7 };    Compare(numbers1, numbers2); |

Здесь подсчет суммы вынесен в локальную функцию Sum, которая принимает массив и возвращает его сумму. И далее в рамках метода Compare мы сможем ее использовать для вычисления суммы массива. При этом неважно, определена локальная функция до или после использования. Но вне ее метода локальная функция не может использоваться.

### Статические локальные функции

Локлальные функции могут быть статическими. Такие функции определяются с помощью ключевого слова **static**. Их особенностью является то, что они не могут обращаться к переменным окружения, то есть метода, в котором статическая функция определена.

Сначала определим локальную функцию, которая имеет доступ окружению:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | int Sum(int[] numbers)  {  int limit = 0;  int result = 0;  foreach (int number in numbers)  {  if (IsPassed(number)) result += number;  }  return result;    bool IsPassed(int number)  {  return number > limit;  }  }    int[] numbers1 = { -3, -2, -1, 0, 1, 2, 3 };  int[] numbers2 = { 3, -4, 5, -6, 7 };    Console.WriteLine(Sum(numbers1));  Console.WriteLine(Sum(numbers2)); |

Здесь функция Sum вычисляет сумму чисел массива, которые соответствуют условию в локальной функции IsPassed(). Эта локальная функция проверяет, больше ли переданное число чем значение переменной limit, определенной в методе Sum. То есть локальная функция IsPassed может обращаться к данным определенным в окружающей функции Sum.

Теперь сделаем функцию IsPassed статической:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | int Sum(int[] numbers)  {  int result = 0;  int limit = 0;  foreach (int number in numbers)  {  if (IsPassed(number, limit)) result += number;  }  return result;    static bool IsPassed(int number, int lim)  {  //return number > limit; // Ошибка: метод IsPassed не имеет доступа к переменной limit  return number > lim;  }  } |

Модификатор **static** указывается перед типом локальной функции. Теперь функция IsPassed не может обращаться к переменной limit, и в этом случае нам надо либо передать это значение в виде параметра, либо определить переменную limit непосредственно в локальной функции.