Рязанский станкостроительный колледж РГРТУ

Основы алгоритмизации и программирования

Тема 5. Модульное программирование. Библиотеки подпрограмм.

Рязань 2020

Оглавление

Библиотеки подпрограмм

Общие сведения

Модульное программирование — это такой способ программирования, при котором вся программа разбивается на группу компонентов, называемых модулями, причем каждый из них имеет свой контролируемый размер, четкое назначение и детально проработанный интерфейс с внешней средой. Единственная альтернатива модульности — монолитная программа, что, конечно, неудобно.

При модульном программировании создаются библиотеки функций, которые расширяют возможности языка.

Например, мы хотим использовать работу с файлами с помощью классов ***StreamWriter*** и ***StreamReader*** для этого, нужно подключить соответствующую библиотеку ***uses System.IO***.

Соответственно мы можем создавать свои модули (библиотеки функций) и использовать их в разрабатываемых программах.

Основные термины

Библиотека – это отдельный программный компонент, который обычно содержит несколько модулей (функции) связанных общим назначением. В C# библиотека представляет собой класс или классы, в который упакованы модули (функции). Каждая библиотека создается в своем пространстве имен ***namespace***, которое в основной программе указывается через директиву ***using***.

Проект - это отдельная программная единица, которая содержит множество элементов для реализации программного продукта определенного направления.

Решение – агрегирующая единица, содержащая несколько проектов, связанных между собой. Окно обозреватель решений показывает все проекты, собранные в данном решении.

Алгоритм создания и использования библиотеки модулей в одном решении с прикладной программой.

1. Создайте проект прикладной программы, любым способом, например:
   - В меню ***Файл*** последовательно выберите ***Создать***, а затем ***Проект.***
   - Из списка шаблонов выберите ***Консольное приложение (.NET Framework).*** Фильтр – C#, Windows, Консоль.
   - Укажите название проекта и место его размещения, при этом откроется заготовка программы.

```
namespace пример1
{
    Ссылок: 0
    class Program
    {
        Ссылок: 0
        static void Main(string[] args)
        {

        }
    }
}
```

Рисунок 1. Заготовка программы

2. Создайте проект библиотеки модулей:
   - В меню ***Файл*** последовательно выберите ***Добавить***, а затем ***Создать проект.***
   - Из списка шаблонов выберите ***Библиотека классов (.Net Framework).*** Фильтр – C#, Windows, Библиотека.
   - Укажите название проекта и место его размещения, при этом откроется заготовка библиотеки.

```
namespace ClassLibrary1
{
    Ссылок: 0
    public class Class1
    {
    }
}
```

Рисунок 2. Заготовка библиотеки

3. Создайте библиотеку модулей:
   - Измените на более подходящее имя пространства имен ***namespace ClassLibrary1.***
   - Измените на более подходящее имя класса ***Class1.***
   - Напишите необходимые модули (функции).
   - Не забудьте установить модификаторы функций ***public*** - для доступа к функции вне класса и ***static*** – для использования функции без создания объекта класса.
   - Выполните сборку (компиляцию) библиотеки ***Сборка – Построить ClassLibrary1*** или ***Собрать решение.*** Обратите внимание, что библиотеку запустить на выполнения нельзя и при компиляции в папке \bin\debug создается файл библиотеки *.dll.

```
namespace ClassLibrary1
{
    ссылка: 1
    public class Class1
    {
        //Функция сумма 2 чисел
        //x,y - входные значения
        //Результат функции сумма чисел
        ссылка: 1
        public static int Sum(int x, int y)
        {
            int rez; // Сумма чисел
            rez = x + y;
            return rez;
        } // end Sum

    }
}
```

Рисунок 3. Пример библиотеки

4. Настройте проект прикладной программы для использования библиотеки.
   - В ***обозревателе решений*** найдите **проект прикладной программы** выберите ***Ссылки (References)***, затем выберите ***Добавить ссылку...*** из контекстного меню.
   - В диалоговом окне "***Менеджер ссылок***" разверните ***Проекты*** и выберите ссылку на библиотеку с вашим названием ***ClassLibrary1***
   - В программном коде **прикладной программы** через директиву ***using*** добавьте ссылку на пространство имен в библиотеке для использования коротких имен модулей. Например, ***using ClassLibrary1***.

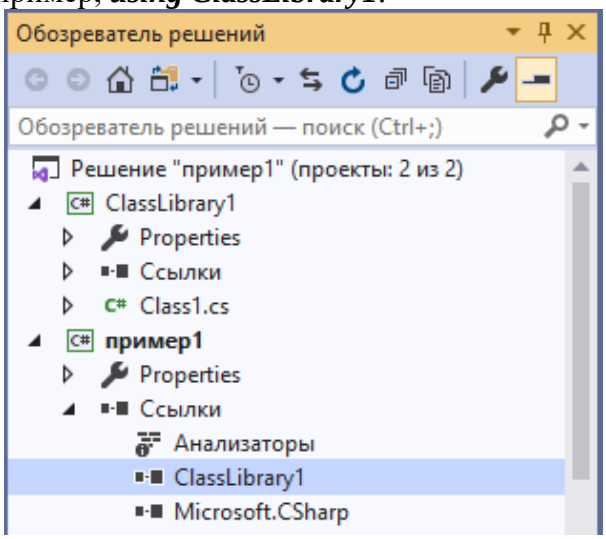

Рисунок 4. Добавление ссылки на библиотеку

5. Используйте функции библиотеки в прикладной программе.

```
using ClassLibrary1;

namespace пример1
{
    Ссылок: 0
    class Program
    {
        //31.03.2020 Сидоров С.С. ИСП-21
        //Пример1. Пример использования библиотеки
        Ссылок: 0
        static void Main(string[] args)
        {
            int zn1, zn2, // Исходные числа
                rez; //Сумма чисел
            Console.WriteLine("Введите 2 значения");
            zn1 = Convert.ToInt32(Console.ReadLine());
            zn2 = Convert.ToInt32(Console.ReadLine()); ;
            rez = Class1.Sum(zn1, zn2);
            Console.WriteLine("Сумма значений = " + rez);
            Console.ReadKey();//Пауза
        }
    }
}
```

Рисунок 5. Пример использования библиотеки функций

Алгоритм использования библиотеки модулей.

В данном случае предполагается, что библиотека уже создана ранее в другом проекте, и мы имеем проверенный и откомпилированный файл библиотеки *.dll.

1. Создайте или откройте проект прикладной программы, любым способом.
2. Настройте проект прикладной программы для использования библиотеки.
   - Скопируйте файл библиотеки ***.dll*** в папку с проектом прикладной программы или запомните путь к файлу ***.dll*** в проекте библиотеке
   - В ***обозревателе решений*** найдите проект прикладной программы выберите ***Ссылки (References)***, затем выберите ***Добавить ссылку...*** из контекстного меню.
   - В диалоговом окне "***Менеджер ссылок***" разверните ***Обзор*** и через кнопку ***Обзор..*** выберите файл библиотеки *.dll.
   - В программном коде формы прикладной программы через директиву ***Using*** добавьте ссылку на пространство имен в библиотеке для использования коротких имен модулей. Например, using ***ClassLibrary1***.
3. Используйте функции библиотеки в прикладной программе

```
using ClassLibrary1;

namespace пример1
{
    Ссылок: 0
    class Program
    {
        //31.03.2020 Сидоров С.С. ИСП-21
        //Пример1. Пример использования библиотеки
        Ссылок: 0
        static void Main(string[] args)
        {
            int zn1, zn2, // Исходные числа
                rez; //Сумма чисел
            Console.WriteLine("Введите 2 значения");
            zn1 = Convert.ToInt32(Console.ReadLine());
            zn2 = Convert.ToInt32(Console.ReadLine()); ;
            rez = Class1.Sum(zn1, zn2);
            Console.WriteLine("Сумма значений = " + rez);
            Console.ReadKey();//Пауза
        }
    }
}
```

Рисунок 6. Пример использования библиотеки функций

Практическая работа № 15.

Создание библиотеки подпрограмм

Задание. Подготовить библиотеку подпрограмм. Использовать подпрограммы для решения поставленной задачи. Оформить программу комментариями.

1. Ввести целое число. Определить:
   а) является ли оно чётным;
   б) оканчивается ли оно цифрой 7;
   в) оканчивается ли оно нечётной цифрой.
2. Ввести двузначное число. Определить:
   а) какая из его цифр больше: первая или вторая;
   б) одинаковы ли его цифры?
   в) кратна ли трём сумма его цифр;
3. Ввести трёхзначное число X. Определить:
   а) является ли сумма его цифр двузначным числом;
   б) больше ли число, введенное А произведение цифр числа X;
   в) кратна ли пяти сумма его цифр;
4. Ввести трёхзначное число.
   а) верно ли, что все его цифры одинаковые?
   б) является ли произведение его цифр трёхзначным числом;
   в) кратна ли сумма его цифр введенному числу А.
5. Ввести четырёхзначное число. Определить:
   а) равна ли сумма двух первых его цифр сумме двух его последних цифр;
   б) кратна ли трём сумма его цифр;
   в) кратно ли произведение его цифр введенному числу А.
6. Ввести натуральное число.
   а) Верно ли, что оно заканчивается нечетной цифрой?
   б) Верно ли, что оно заканчивается четной цифрой?
   в) входят ли в него цифры 4 или 7;
7. Ввести четырехзначное число. Определить:
   а) входит ли в него цифра 4;
   б) входит ли в него цифра В.
   в) кратно ли четырём произведение его цифр;
8. Определить, является ли введенное шестизначное число:
   а) счастливым. (Счастливым называют такое шестизначное число, у которого сумма его первых трех цифр равна сумме его последних трех цифр).
   б) является ли сумма его цифр трехзначным числом;
9. Составить программу, которая:
   а) уменьшает первое веденное число в два раза, если оно больше второго введенного числа по абсолютной величине.
   б) Если квадратный корень из второго введенного числа меньше первого введенного числа, то увеличить второе число в пять раз.
   в) Определить сумму из двух веденных чисел, которые кратны трем.
10. Ввести три целых числа.
    а) Подсчитать кол-во чисел, которые являются четными.
    б) Возвести в квадрат те из них, значения которых неотрицательны.
    в) Найти сумму тех чисел, которые больше пяти.
11. Ввести целое число. Определить:
    а) является ли оно нечётным;
    б) оканчивается ли оно цифрой 1;

в) начинается ли оно с цифры 2.

12. Ввести двузначное число. Определить:
    а) определить больше ли первая цифра чем вторая в 2 раза;
    б) одинаковы ли его цифры?
    в) кратна ли число 3;
13. Ввести трёхзначное число X. Определить:
    а) является ли сумма его цифр четным числом;
    б) больше ли число, введенное А сумме цифр числа X;
    в) Вывести цифры числа по возрастанию;
14. Ввести трёхзначное число.
    а) верно ли, что все его цифры разные одинаковые?
    б) является ли сумма его цифр двухзначным числом;
    в) кратна ли сумма его цифр введенному числу А.
15. Ввести четырёхзначное число. Определить:

а) равна ли произведение двух первых его цифр сумме двух его последних цифр;

б) кратна ли четырем сумма его цифр;

в) кратно ли произведение его цифр, каждой цифре числа.

16. Определить, является ли введенное шестизначное число:

а) счастливым. (Счастливым называют такое шестизначное число, у которого сумма его первых трех цифр равна сумме его последних трех цифр).

б) является ли сумма его цифр трехзначным числом;

в) найти сумму четных цифр числа.

17. Ввести 2 числа:

а) уменьшить первое веденное число на два, если оно больше второго введенного числа по абсолютной величине.

б) Если квадрат второго введенного числа меньше первого введенного числа, то увеличить второе число в пять раз.

в) Определить сумму из двух веденных чисел, которые кратны трем.

Контрольные вопросы

1. С какой целью оформляются и где описываются процедуры и функции?
2. В каком случае подпрограмма оформляется как процедура и в каком случае как функция?
3. Каков синтаксис описания процедуры? Какой вид имеет заголовок процедуры?
4. Каков синтаксис описания функции?
5. Каковы особенности реализации функций?
6. Как выполняются обращения к процедуре и к функции?
7. Какие переменные подпрограмм называются локальными и какова область их действия? Какие переменные называются глобальными?
8. Какие параметры используются для передачи данных между подпрограммами, какими способами осуществляется эта передача?
9. Что такое формальные параметры подпрограммы и как они описываются? Что такое фактические параметры подпрограммы?
10. Какие существуют виды формальных параметров?
11. Как передаются в подпрограмму параметры-значения, что может использоваться в качестве фактического параметра для параметра-значения?
12. Как передаются в подпрограмму параметры-переменные, что может использоваться в качестве фактического параметра для параметра-переменной?
13. Как передаются в подпрограмму параметры-константы, что может использоваться в качестве фактического параметра для параметра-константы?
14. Каким образом можно осуществить досрочный выход из подпрограммы?
15. Существуют ли подпрограммы без параметров?
16. Какое количество значений возвращает функция?
17. Как определить тип значения, возвращаемый функцией?
18. Могут ли фактические параметры быть выражениями?
19. Сколько элементов должен содержать список фактических параметров?
20. Сколько элементов может содержать список формальный параметров?
21. Каково соответствие между фактическими и формальными параметрами?
22. Могут ли фактические параметры быть именами переменных?
23. Какие переменные называются локальными?
24. Что такое область видимости переменной?
25. Может ли имя локальной переменной совпадать с глобальной?
26. Может ли в основной программе функция вызываться внутри выражений?
27. Наличие какого оператора необходимо для возвращения значения из функции в вызывающую программу?