**Решения на задачите от глава 01 на книгата "Въведение в програмирането със C#"**

Предлагаме ви решения на задачите от [книгата "Въведение в програ­мирането със C#"](http://www.introprogramming.info), заедно с анализ на задачата, описание на използва­ните идеи, алгоритми, подходи за решаване и тестове.

**Авторски колектив**

**Кирил Попов**

Контакти: kpopov.soft@gmail.com

**Димитър Тачев**

Контакти: cska\_\_forever@abv.bg

**Мариан Маринов**

Контакти: lezetac@gmail.com

**Наталия Илиева**

Контакти: ilieva.natalia@gmail.com

**Задача 1.** ***Първи стъпки с “Microsoft Visual Studio” и “Microsoft Developer Network (MSDN) Library Documentation”***

Visual Studio 2010 (VS 1020) е мощна интегрирана среда за разработка на софтуерни приложения за Windows и за платформата .NET Framewоrk. VS 2010 поддържа различни езици за програмиране (например C#, VB.NET и C++) и различни технологии за разработка на софтуер (Win32, COM, ASP.NET, ADO.NET Entity Framework, Windows Forms, WPF, Silverlight и още десетки други Windows и .NET технологии). VS 2010 предоставя мощна интегрирана среда за писане на код, компилиране, изпълнение, дебъгване и тестване на приложения, дизайн на потребителски интерфейс (форми, диалози, уеб страници, визуални контроли и други), моделиране на данни, моделиране на класове, изпълнение на тестове, пакетиране на приложения и стотици други функции.

Библиотеката MSDN Library представлява официалната документация на Microsoft за всички техни продукти за разработчици и софтуерни технологии. В частност документацията за .NET Framework е част от MSDN Library и е достъпна в Интернет тук: [MSDN](http://msdn.microsoft.com/en-us/library/ms229335(VS.100).aspx).

|  |
| --- |
| **Задача 2. System.Console class** |
| **Условие**  Да се намери описанието на класа System.Console в стандартната .NET API документация (MSDN Library) |
|  |
|  |
| The console is an operating system window where users interact with the operating system or a text-based console application by entering text input through the computer keyboard, and reading text output from the computer terminal. For example, in Windows the console is called the command prompt window and accepts MS-DOS commands. The **Console** class provides basic support for applications that read characters from, and write characters to, the console. |

**Задача 3.** Да се намери описанието на метода**System.Console.WriteLine()**сразлич­ните негови възможни параметри в MSDN Library.

<http://msdn.microsoft.com/en-us/library/system.console.writeline.aspx>

*Console.WriteLine Method*

Writes the specified data, followed by the current line terminator, to the standard output stream.This member is overloaded.

* WriteLine Writes the current line terminator to the standard output stream.
* WriteLine(Boolean) Writes the text representation of the specified Boolean value, followed by the current line terminator, to the standard output stream.
* WriteLine(Char) Writes the specified Unicode character, followed by the current line terminator, value to the standard output stream.
* WriteLine(Char()) Writes the specified array of Unicode characters, followed by the current line terminator, to the standard output stream.
* WriteLine(Decimal) Writes the text representation of the specified Decimal value, followed by the current line terminator, to the standard output stream.
* WriteLine(Double) Writes the text representation of the specified double-precision floating-point value, followed by the current line terminator, to the standard output stream.
* WriteLine(Int32) Writes the text representation of the specified 32-bit signed integer value, followed by the current line terminator, to the standard output stream.
* WriteLine(Int64) Writes the text representation of the specified 64-bit signed integer value, followed by the current line terminator, to the standard output stream.
* WriteLine(Object) Writes the text representation of the specified object, followed by the current line terminator, to the standard output stream.
* WriteLine(Single) Writes the text representation of the specified single-precision floating-point value, followed by the current line terminator, to the standard output stream.
* WriteLine(String) Writes the specified string value, followed by the current line terminator, to the standard output stream.
* WriteLine(UInt32) Writes the text representation of the specified 32-bit unsigned integer value, followed by the current line terminator, to the standard output stream.
* WriteLine(UInt64) Writes the text representation of the specified 64-bit unsigned integer value, followed by the current line terminator, to the standard output stream.
* WriteLine(String, Object) Writes the text representation of the specified object, followed by the current line terminator, to the standard output stream using the specified format information.
* WriteLine(String, Object()) Writes the text representation of the specified array of objects, followed by the current line terminator, to the standard output stream using the specified format information.
* WriteLine(Char(), Int32, Int32) Writes the specified subarray of Unicode characters, followed by the current line terminator, to the standard output stream.
* WriteLine(String, Object, Object) Writes the text representation of the specified objects, followed by the current line terminator, to the standard output stream using the specified format information.
* WriteLine(String, Object, Object, Object) Writes the text representation of the specified objects, followed by the current line terminator, to the standard output stream using the specified format information.
* WriteLine(String, Object, Object, Object, Object) Writes the text representation of the specified objects and variable-length parameter list, followed by the current line terminator, to the standard output stream using the specified format information.

|  |  |
| --- | --- |
| Задача 4. Hello C#! | |
| **Условие**  Да се компилира и изпълни примерната програма от примерите в тази глава през командния ред (конзолата) и с помощта на Visual Studio. | |
| **Описание на входа**  Няма. | |
| **Описание на изхода**  Изходните данни трябва да се изведат през командния ред (конзолата). Изходът представлява изречението **„Hello C#!“.** | |
| **Анализ на задачата**  Използва се системният метод **System.Console.WriteLine(string value)**, чрез който се извежда “Hello C#!” през командния ред. | |
| **Решение (сорс код)** | |
| using System;  class HelloCSharp  {  static void Main(string[] args)  {  System.Console.WriteLine("Hello C#!");  }  } | |
| **Тестове**  Единствен тест ще проверява дали се изписва желаното от нас изречение, а именно “**Hello C#!**”. | |
| **Вход** | **Изход** |
| Няма | Hello C#! |

|  |  |
| --- | --- |
| **Задача 5. *Поздрав „Добър ден!“***. | |
| **Условие**  Да се модифицира примерната програма от книгата, така че да изписва поздравлението: "Добър ден!". | |
| **Описание на входа**  Няма. | |
| **Описание на изхода**  Изходните данните трябва да се изведат на стандартния изход (конзолата).  На единствения ред на стандартния изход трябва да изведете поздравлението „Добър ден!“. | |
| **Анализ на задачата**  Чрез статичния системен метод:  System.Console.WriteLine(string **value**) на конзолата се извежда дадения поздрав. | |
| **Решение (сорс код)** | |
| using System;  class HelloCSharp  {      static void Main()      {          System.Console.WriteLine("Добър ден!");      }  } | |
| **Тестове**  Само един тест ще проверява дали на конзолата се изписва „ Добър ден!“. | |
| **Вход** | **Изход** |
| Няма | Добър Ден! |

|  |  |
| --- | --- |
| Задача 6.  Програма, която изписва вашето име и фамилия на конзолата. | |
| **Условие**  Напишете програма, която изписва вашето име и фамилия на конзолата. | |
| **Описание на входа**  Няма. | |
| **Описание на изхода**  Изхода трябва да отпечатва на екрана името на потребителя. | |
| **Анализ на задачата**  няма | |
| **Решение (сорс код)** | |
| using System;  class PrintFirstAndLastNameApp  {  static void Main(string[] args)  {  Console.WriteLine("Kiril Popov");  }  } | |
| **Тестове**  В задачата се ползва единствено Console.WriteLine(string), поради тази причина тестове не са необходими. Появата на IOException би значела сериозна повреда в системата. | |
| **Вход** | **Изход** |
| (липсва) | Kiril Popov |

|  |  |
| --- | --- |
| **Задача 7.  Отпечатване на 1, 101 и 1001** | |
| **Условие**  Напишете програма, която извежда на конзолата числата 1, 101 и 1001 всяко на нов ред. | |
| **Описание на входа**  няма | |
| **Описание на изхода**  Изхода трябва да отпечатва на екрана  „1  101  1001“ | |
| **Анализ на задачата**  няма | |
| **Решение (сорс код)** | |
| using System;  class Program  {  static void Main()  {  Console.WriteLine(1);  Console.WriteLine(101);  Console.WriteLine(1001);  }  } | |
| **Тестове**   * Проверка дали на конзолата се извеждат исканите три числа. | |
| **Вход** | **Изход** |
| (липсва) | 1  101  1001 |

|  |  |
| --- | --- |
| **Задача 8. Текуща дата и час.** | |
| **Условие**  Напишете програма, която извежда на конзолата текущата дата и час. | |
| **Описание на входа**  Няма. | |
| **Описание на изхода**  Изходът представлява един ред от конзолата, на който се изписват текущата дата и час. | |
| **Анализ на задачата**  Декларираме променливата **currentTime** от тип **DateTime**, на която присвояваме свойството **DateTime.Now**. Това свойство връща, като резултат текущата дата и час.  След това използваме системният метод **System.Console.WriteLine()**, за да изведем стойността на променливата **currentTime** на конзолата. | |
| **Решение (сорс код)** | |
| using System;  class CurrentDateTime  {  static void Main(string[] args)  {  DateTime currentTime = DateTime.Now;  System.Console.WriteLine(currentTime);  }  } | |
| **Тестове**  Единствена проверка за съвпадение между изписаните дата и час и текущите такива. | |
| **Вход** | **Изход** |
| Няма | Текуща дата и час |

|  |  |
| --- | --- |
| **Задача 9. *Корен Квадратен.*** | |
| **Условие**  Напишете програма, която извежда корен квадратен от числото 12345. | |
| **Описание на входа**  Няма | |
| **Описание на изхода**  Изходните данните трябва да се изведат на стандартния изход (конзолата).  На единствения ред на стандартния изход трябва да изведете корена на числото 12345 (число от тип ***“double”*** с десетичен разделител запетая ***“,”***). | |
| **Анализ на задачата**  Чрез статичните системни методи:  System.Console.WriteLine(string **value**) и System.Math.Sqrt(**double** **d**) на конзолата се извежда корена на числото 12345. | |
| **Решение (сорс код)** | |
| using System;    class SQRT  {      static void Main(string[] args)      {          Console.WriteLine(System.Math.Sqrt(12345));      }  } | |
| **Тестове**  Само един тест ще проверява дали на конзолата се изписва корена на числото 12345. | |
| **Вход** | **Изход** |
| Няма | 111,108055513541 |

|  |  |
| --- | --- |
| **Задача 10.  Програма, която извежда първите 100 члена на редицата 2, -3, 4, -5, 6, -7, 8.** | |
| **Условие**  Напишете програма, която извежда първите 100 члена на редицата 2, -3, 4, -5, 6, -7, 8. | |
| **Описание на входа**  Няма. | |
| **Описание на изхода**  Изхода трябва да отпечатва на екрана първите 100 члена на редицата 2, -3, 4, -5, 6,  -7, 8. | |
| **Анализ на задачата**  няма | |
| **Решение (сорс код)** | |
| using System;  class PrintSequenceApp  {    static void Main(string[] args)  {  for (int i = 0; i < 100; i++)  {  if (i % 2 == 0)  {  Console.Write("{0,4} ", i + 2);  }  else  {  Console.Write("{0,4} ", -(i + 2));  }  }  Console.WriteLine();  }  } | |
| **Тестове**   * Проверка дали елементите, които се извеждат на конзолата, са точно исканите 100 числа. | |
| **Вход** | **Изход** |
| (няма) | 2 -3 4 -5 6 -7 8 -9 10 -11 12 -13 14 -15 16 -17  18 -19 20 -21 22 -23 24 -25 26  -27 28 -29 30 -31 32 -33  34 -35 36 -37 38 -39 40 -41 42 -43 44 -45 46 -47 48 -49  50 -51 52 -53 54 -55 56 -57 58 -59 60 -61 62 -63 64 -65  66 -67 68 -69 70 -71 72 -73 74 -75 76 -77 78 -79 80 -81  82 -83 84 -85 86 -87 88 -89 90 -91 92 -93 94 -95 96 -97  98 -99 100 -101 |

|  |  |
| --- | --- |
| **Задача 11. Възрастта ви след 10 години** | |
| **Условие**  Направете програма, която прочита от конзолата вашата възраст и изписва (също на конзолата) каква ще бъде вашата възраст след 10 години. | |
| **Описание на входа**  На входа на програмата се подава едно единствено цяло число- възрастта на потребителя. | |
| **Описание на изхода**  При правилно въведени данни, на изхода получаваме числото, показващо възрастта на потребителя след 10 г. | |
| **Анализ на задачата**  Възрастта на потребителя ще се чете и превръща в цяло число. Ако това число е недопустима възраст(по-малко от 1 или по-голямо от 120), ще се отпечата съобщение за грешка. | |
| **Решение (сорс код)** | |
| using System;  class Program  {  static void Main()  {  int age;  string ageString;  ageString = Console.ReadLine();  try  {  age = int.Parse(ageString);  if (age <= 0 || age > 120)  {  throw new OverflowException();  }  Console.WriteLine(age + 10);  }  catch (ArgumentNullException)  {  Console.WriteLine("Age missing.");  }  catch (FormatException)  {  Console.WriteLine("Not a valid number.");  }  catch (OverflowException)  {  Console.WriteLine("Not a valid age.");  }  }  } | |
| **Тестове**  Програмата ще се тества със следните входове:   * Нормална възраст (примерно 35) * Отрицателно число * Дробно число * Знаци, които не са число * Числа в различна бройна система (вместо #, въвеждаме 0#, 0х#) * Нереалнo голяма възраст * Огромно число * Липсващ вход * Въвеvдане на възрастта с табулации/интервали около него | |
| **Вход** | **Изход** |
| 35 | 45 |
| **Вход** | **Изход** |
| -1 | Not a valid age. |
| **Вход** | **Изход** |
| 2.5 | Not a valid number. |
| **Вход** | **Изход** |
| asdf | Not a valid number. |
| **Вход** | **Изход** |
| 0x5 | Not a valid number. |
| **Вход** | **Изход** |
| 500 | Not a valid age. |
| **Вход** | **Изход** |
| 10000000000000000000000000000000000000000000000000000000000000000000000000000 | Not a valid age. |
| **Вход** | **Изход** |
| (липсва) | Not a valid number. |
| **Вход** | **Изход** |
| 20 | 30 |

**Задача 12:** Опишете разликите между C# и .NET Framework.

* What is C#?

**C#** (pronounced *see sharp*) is a [multi-paradigm programming language](http://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [strong typing](http://en.wikipedia.org/wiki/Strong_typing), [imperative](http://en.wikipedia.org/wiki/Imperative_programming), [declarative](http://en.wikipedia.org/wiki/Declarative_programming), [functional](http://en.wikipedia.org/wiki/Functional_programming), [generic](http://en.wikipedia.org/wiki/Generic_programming), [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming) ([class-based](http://en.wikipedia.org/wiki/Class_(computer_science))), and [component-oriented](http://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. It was developed by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) within its [.NET](http://en.wikipedia.org/wiki/.NET_Framework) initiative and later approved as a standard by [Ecma](http://en.wikipedia.org/wiki/Ecma_International) (ECMA-334) and [ISO](http://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO/IEC 23270:2006). C# is one of the programming languages designed for the [Common Language Infrastructure](http://en.wikipedia.org/wiki/Common_Language_Infrastructure).

C# is intended to be a simple, modern, general-purpose, object-oriented programming language.[[6]](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)#cite_note-ECMA-334-6) Its development team is led by [Anders Hejlsberg](http://en.wikipedia.org/wiki/Anders_Hejlsberg). The most recent version is [C# 4.0](http://en.wikipedia.org/wiki/C_Sharp_4.0), which was released on April 12, 2010.

* What is .NET Framework?

The **.NET Framework** (pronounced *dot net*) is a [software framework](http://en.wikipedia.org/wiki/Software_framework) that runs primarily on [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows). It includes a large [library](http://en.wikipedia.org/wiki/Base_Class_Library) and provides language interoperability (each language can use code written in other languages) across several [programming languages](http://en.wikipedia.org/wiki/Programming_language). Programs written for the .NET Framework execute in a [software](http://en.wikipedia.org/wiki/Software) environment (as contrasted to [hardware](http://en.wikipedia.org/wiki/Hardware) environment), known as the [Common Language Runtime](http://en.wikipedia.org/wiki/Common_Language_Runtime) (CLR), an [application virtual machine](http://en.wikipedia.org/wiki/Virtual_machine#Process_virtual_machines) that provides important services such as security, [memory management](http://en.wikipedia.org/wiki/Memory_management), and [exception handling](http://en.wikipedia.org/wiki/Exception_handling). The class library and the CLR together constitute the .NET Framework.

The .NET Framework's [Base Class Library](http://en.wikipedia.org/wiki/Base_Class_Library) provides [user interface](http://en.wikipedia.org/wiki/User_interface), [data access](http://en.wikipedia.org/wiki/Data_access), [database connectivity](http://en.wikipedia.org/wiki/Database_connection), [cryptography](http://en.wikipedia.org/wiki/Cryptography), [web application](http://en.wikipedia.org/wiki/Web_application) development, numeric [algorithms](http://en.wikipedia.org/wiki/Algorithm), and [network communications](http://en.wikipedia.org/wiki/Computer_networking). Programmers produce software by combining their own [source code](http://en.wikipedia.org/wiki/Source_code) with the .NET Framework and other libraries. The .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces a popular [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) largely for .NET software called [Visual Studio](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio).

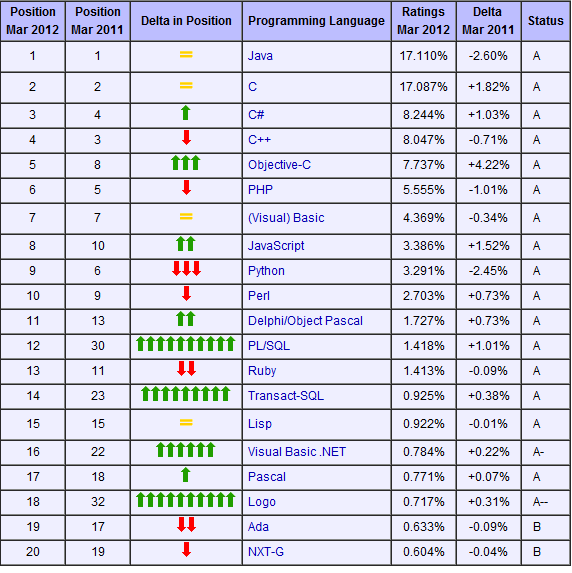
**Sources:**

<http://en.wikipedia.org/wiki/.NET_Framework>

<http://en.wikipedia.org/wiki/C_Sharp_(programming_language)>

**Задача 13.** Направете списък с най- популярните програмни езици. С какво те се различават от C#.

По информация на TIOBE най-популярните 5 езикa за програмиране за месец Март 2011та са:



С# е език от високо ниво, който прилича на Java и C++ и донякъде на езици като Delphi, VB.NET и С, но въпреки големите си прилики с Java той е доста по богат и бърз, поради което често бива предпочитан. За разлика от С# и .NET Framework езика и платформата на Java са проекти с отворен код, в който участва цялата общност от софтуерни фирми, организации, и индивидуални разработчици. Стандартите, спецификациите и всички новости в Java света се развиват чрез работни групи, съставени от цялата Java общност, а не от една единствена фирма както при Microsoft продуктите.

За разлика от С и С++ за който се смята, че са доста примитивни, остарели и умиращи езица за програмиране С# е съвременен и изключително прост и лесен за научаване език. Поостарелият С все още е популярен, защото е незаменим при писане на операционни системи, хардуерни драйвери или за програмиране на промишлени контролери. С++ пък поради по-голямата си близост до С е подходящ при писане на отделни приложения, които изискват много близка работа с хардуера или имат специални изисквания за бързодействието. За всички други случаи С# е в пъти по-добър, което най-вече се дължи на това че е обектно-ориентиран и позволява бърза и лесна работа с обектите от реалния свят.

С# намира приложение и при реализация на големи и сложни софтуерни системи където езици като PHP изпитват големи трудности. РHP е език удобен за писане на малки Уеб-базирани проекти.