C# Chapter 11

1. Write a program that reads from the console year and checks if it is high.

static  void  Main ( string [] args)  
{  
    Console.Write ( "Enter year:" );  
    int  year = Int32.Parse (Console.ReadLine ());  
    if  ((year% 4) == 0) && ((year% 100)! = 0) || ((year% 400) == 0)))  
        Console.WriteLine ( "{0} is leap year. " , year);  
    else  
        Console.WriteLine ( "{0} isn't a leap year." , year);  
}

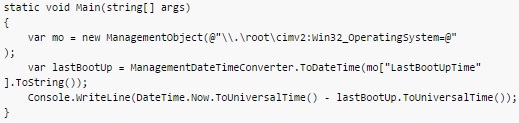
1. Write a program that generates and prints on the console 10 random numbers in the range [100, 200].

static  void  Main ( string [] args)  
{  
    Random r =  new  Random ();  
    for  ( int  i = 0; i <10; i ++)  
        Console.WriteLine (r.Next (100, 201));  
}

1. Write a program that displays on the console which day of the week is today.

static  void  Main ( string [] args)  
{  
    Console.Write ( "Enter year:" );  
    int  year = Int32.Parse (Console.ReadLine ());  
    Console.Write ( "Enter month:" );  
    int  month = Int32.Parse (Console.ReadLine ());  
    Console.Write ( "Enter day:" );  
    int  day = Int32.Parse (Console.ReadLine ());  
      
    DateTime dateValue =  new  DateTime (year, month, day);  
    Console.WriteLine (dateValue.ToString ( "ddd" ));  
}

1. Write a program that outputs to standard output the number of days, hours, and minutes that have elapsed since the computer was turned on until the program ran. Use the class for the conversion Environment.



1. Write a program that finds the hypotenuse of a right triangle on two legs. Enter the lengths of the legs from the standard input, and use the class methods to calculate the hypotenuse Math.

static  void  Main ( string [] args)  
{  
    Console.Write ( "First side:" );  
    int  a = Int32.Parse (Console.ReadLine ());  
    Console.Write ( "Second side:" );  
    int  b = Int32.Parse (Console.ReadLine ());  
      
    Console.Write ( "Hypotenuse is:"  + Math.Sqrt (Math.Pow (a, 2) + Math.Pow (b, 2)));  
}

1. Write a program that calculates the face of a triangle by:  
   a. the lengths of its three sides;  
   b. the length of one of the sides and the height to it;  
   c. the lengths of two of its sides and the angle between them in degrees.

using System;

namespace Chapter\_11\_Solution\_6

{

class Program

{

public static void byThreeSides()

{

Console.Write("\nEnter a: ");

float a = float.Parse(Console.ReadLine());

Console.Write("Enter b: ");

float b = float.Parse(Console.ReadLine());

Console.Write("Enter c: ");

float c = float.Parse(Console.ReadLine());

float p = (a + b + c) / 2;

Console.WriteLine("S is: {0}", (float)(Math.Sqrt(p \* (p - a) \* (p - b) \* (p - c))));

Console.ReadLine();

}

public static void bySideAltitude()

{

Console.Write("\nEnter a: ");

float a = float.Parse(Console.ReadLine());

Console.Write("Enter h(a): ");

float b = float.Parse(Console.ReadLine());

Console.WriteLine("S is: {0}", (a \* b) / 2);

Console.ReadLine();

}

public static void byTwoSidesAngle()

{

Console.Write("\nEnter a: ");

float a = float.Parse(Console.ReadLine());

Console.Write("Enter b: ");

float b = float.Parse(Console.ReadLine());

Console.Write("Enter sine: ");

float c = float.Parse(Console.ReadLine());

Console.WriteLine("S is: {0}", (a \* b \* Math.Sin(c)) / 2);

Console.ReadLine();

}

static void Main(string[] args)

{

byte choice;

do

{

Console.Clear();

Console.WriteLine("Calculate the area of a triangle by given:");

Console.WriteLine("1.Three side lengths.");

Console.WriteLine("2.Length of one side and it's altitude.");

Console.WriteLine("3.Length of two sides and the angle between them.");

Console.WriteLine("4.Exit.");

Console.Write("Enter choice from 1 to 4: ");

choice = System.Convert.ToByte(Console.ReadLine());

switch (choice)

{

case 1: byThreeSides(); break;

case 2: bySideAltitude(); break;

case 3: byTwoSidesAngle(); break;

}

} while (choice != 4);

}

}

}

1. Define your own space Chapter11and place it in both classes Catand Sequencewe used the examples of current theme. Make another own space named Chapter11.Examplesin it do the class that invokes classes Catand Sequence.

using System;

namespace Chapter\_11\_Solution\_7.Folder

{

class Sequence

{

private static int currentValue = 0;

private Sequence()

{

}

public static int NextValue()

{

currentValue++;

return currentValue;

}

}

}

1. Write a program that creates 10 objects of type Cat, gives them names of the type CatNwhere the Nsequence number of the object is unique, and finally calls the method SayMiau()of each of them. Use the already defined Chapter11 space for the implementation.

using System;

namespace Chapter\_11\_Solution\_7.Folder

{

class Cat

{

private string name;

private string color;

public Cat(string name, string color)

{

this.name = name;

this.color = color;

}

public void sayMiau()

{

Console.WriteLine("{0}: Miauu", name);

}

}

}

1. Write a program that calculates the number of working days between today's date and another date after today's (inclusive). Working days are all days without Saturday and Sunday, which are not official holidays, and exceptionally Saturday may be a working day when weekends are worked around the holidays. The program must keep a list of pre-set public holidays, as well as a list of pre-set working Saturdays.

using System;

namespace Chapter\_11\_Solution\_9

{

class Program

{

static void Main(string[] args)

{

DateTime[] holidays = new DateTime[]

{

new DateTime(2014, 12, 24),

new DateTime(2014, 12, 25),

new DateTime(2015, 01, 01),

new DateTime(2015, 01, 02),

new DateTime(2015, 03, 02),

new DateTime(2015, 03, 03),

new DateTime(2015, 04, 10),

new DateTime(2015, 04, 13),

new DateTime(2015, 05, 01),

new DateTime(2015, 05, 06),

new DateTime(2015, 09, 21),

new DateTime(2015, 09, 22),

new DateTime(2015, 12, 24),

new DateTime(2015, 12, 25),

};

DateTime[] workSaturdays = new DateTime[]

{

new DateTime(2015, 01, 24),

new DateTime(2015, 03, 21),

new DateTime(2015, 09, 12),

new DateTime(2015, 12, 12),

}

int workingDays = 0;

Console.Write("Enter end date (YYYY/MM/DD): ");

DateTime endDate = System.Convert.ToDateTime(Console.ReadLine());

DateTime now = DateTime.Now;

do

{

now = now.AddDays(1);

if((now.DayOfWeek >= DayOfWeek.Monday) && (now.DayOfWeek <= DayOfWeek.Friday))

workingDays++;

foreach (var i in holidays)

if (i.Date == now.Date)

workingDays--;

foreach (var i in workSaturdays)

if (i.Date == now.Date)

workingDays++;

} while (now.Date != endDate.Date);

Console.WriteLine("{0} working days.", workingDays);

}

}

}

1. A sequence of positive integers is given, written one after the other as a string, separated by a space. Write a program that calculates their amount. Example: "43 68 9 23 318" = 461.

using System;

namespace Chapter\_11\_Solution\_10

{

class Program

{

static void Main(string[] args)

{

int result = 0;

Console.Write("Enter numbers: ");

string inputNumbers = Console.ReadLine();

string[] splitNumbers = inputNumbers.Split(' ');

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

result += Convert.ToInt32(splitNumbers[i]);

Console.WriteLine("Result is: {0}", result);

}

}

}

1. Write a program that generates a random advertisement for a product. Messages should consist of a praise phrase, followed by a praise event, followed by an author (first and second name) and a city, which are selected from pre-prepared lists. For example, let's have the following lists:

using System;

namespace Chapter\_11\_Solution\_11

{

class Program

{

static void Main(string[] args)

{

Random rnd = new Random();

string[] firstNames = new string[] { "Diana", "Petia", "Stela", "Elena", "Katia" };

string[] secondNames = new string[] { "Ivanova", "Petrova", "Kirova", "Popova" };

string[] cities = new string[] { "Sofia", "Plovdiv", "Varna", "Ruse", "Dolno konare" };

string[] reviews1 = new string[] { "This product is excelent.", "This is a great product.", "I use this product all the time.", "This is the best product of this category." };

string[] reviews2 = new string[] { "Now I feel better.", "I changed.", "It was a miracle.", "I can't believe it. Now I feel better.", "Try it yourself. I'm very greatful." };

Console.WriteLine("{0} {1}\n-- {2} {3}, {4}.", reviews1[rnd.Next(4)], reviews2[rnd.Next(5)], firstNames[rnd.Next(5)], secondNames[rnd.Next(4)], cities[rnd.Next(5)]);

}

}

}

1. Write a program that calculates the value of a numeric expression specified as a string. The numeric expression consists of:  
   - real numbers, for example 5, 18.33, 3.14159, 12.6;  
   - arithmetic operators: +, -, \*, /(with their standard priorities);  
   - mathematical functions: ln(x), sqrt(x), pow(x,y);  
   brackets for changing the priority of operations:.  
   Note that numeric expressions have priority, such as the expression-1 + 2 + 3 \* 4 - 0.5 = (-1) + 2 + (3 \* 4) - 0.5 = 12.5.

using System;

using System.Linq;

namespace Chapter\_11\_Solution\_12

{

class Program

{

public static string equation;

public static string[] equationArr = new string[50];

public static string[] actions = new string[50];

public static string[] numbers = new string[50];

public static bool change = false;

public static double result = 0;

public static double temp = 0;

public static byte depth = 0;

public static byte brakets = 0;

public static void removeNulls()

{

numbers = numbers.Where(x => !string.IsNullOrEmpty(x)).ToArray();

actions = actions.Where(x => !string.IsNullOrEmpty(x)).ToArray();

}

static void Main(string[] args)

{

Console.WriteLine("Enter equation: ");

equation = Console.ReadLine();

equationArr = equation.Split(' ');

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

{

if (change) actions[i] = equationArr[i];

else numbers[i] = equationArr[i];

change = change ? false : true;

}

removeNulls();

do

{

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

if (i == 0)

if (actions[i] == "\*" || actions[i] == "/")

{

if (actions[i] == "\*")

temp = Convert.ToDouble(numbers[0]) \* Convert.ToDouble(numbers[1]);

else temp += Convert.ToDouble(numbers[0]) / Convert.ToDouble(numbers[1]);

numbers[0] = temp.ToString();

actions[i] = numbers[1] = null;

temp = 0;

removeNulls();

}

else

do

{

if (actions[i] == "\*" || actions[i] == "/")

{

if (actions[i] == "\*") temp = Convert.ToDouble(numbers[i]) \* Convert.ToDouble(numbers[i + 1]);

else temp += Convert.ToDouble(numbers[i]) / Convert.ToDouble(numbers[i + 1]);

numbers[i] = temp.ToString();

actions[i] = numbers[i + 1] = null;

temp = 0;

removeNulls();

if (depth > 0) depth--;

}

foreach (var s in actions)

if (s == "\*" || s == "/") depth++;

} while (depth != 0);

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

Console.WriteLine(actions[i]);

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

Console.WriteLine(numbers[i]);

Console.WriteLine("Continue {0} to +-", result);

Console.ReadLine();

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

if (i == 0)

if (actions[i] == "+" || actions[i] == "-")

{

if (actions[i] == "+")

numbers[0] = (Convert.ToDouble(numbers[0]) + Convert.ToDouble(numbers[1])).ToString();

else numbers[0] = (Convert.ToDouble(numbers[0]) - Convert.ToDouble(numbers[1])).ToString();

actions[i] = numbers[1] = null;

removeNulls();

i--;

}

else

do

{

if (actions[i] == "+" || actions[i] == "-")

{

if (actions[i] == "+")

numbers[i] = (Convert.ToDouble(numbers[i]) + Convert.ToDouble(numbers[i + 1])).ToString();

else numbers[i] = (Convert.ToDouble(numbers[i]) - Convert.ToDouble(numbers[i + 1])).ToString();

actions[i] = numbers[i + 1] = null;

removeNulls();

if (depth > 0) depth--;

}

foreach (var s in actions)

if (s == "+" || s == "-") depth++;

} while (depth != 0);

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

Console.WriteLine("actions {0}",actions[i]);

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

Console.WriteLine("numbers {0}", numbers[i]);

} while (numbers.Length > 1);

Console.WriteLine("Result is {0}", numbers);

}

}

}