**Introduction to programming**

//Problem 14.\* **Current Date and Time**

//Create a console application that prints the current date and time. Find out how in Internet.

using System;

class DateAndTime

{

static void Main()

{

Console.WriteLine(DateTime.Now); // using DateTime

}

}

*DateTime TodayDate = DateTime.Today; DateTime DateOnly = TodayDate.Date;*

*Console.WriteLine("Today is {0}", DateOnly.ToString("d"));*

//Problem 15.\* **Age after 10 Years**

//Write a program to read your birthday from the console and print how old you are now and how old you will be after 10 years.

using System;

class AgeAfter10Years

{

static void Main()

{

Console.Write("Which date were you born (dd.mm.yyyy): ");

DateTime dateOfBirth = Convert.ToDateTime(Console.ReadLine());

DateTime presentYear = DateTime.Now;

TimeSpan ts = presentYear - dateOfBirth;

DateTime Age = DateTime.MinValue.AddDays(ts.Days);

Console.WriteLine("Now I'm at: {0} Years {1} Month {2} Days", Age.Year - 1, Age.Month - 1, Age.Day - 1);

int ageAFterTen = Age.Year + 9;

Console.WriteLine("After 10 Years I'll be at: {0} Years {1} Month {2} Days", ageAFterTen, Age.Month - 1, Age.Day - 1);

}

}

//Problem 16.\* **Print Long Sequence**

//Write a program that prints the first 1000 members of the sequence: 2, -3, 4, -5, 6, -7, …

//You might need to learn how to use loops in C# (search in Internet).

using System;

class PrintLongSequence

{

static void Main()

{

for (int i = 2; i < 1002; i++)

{

if (i % 2 == 0)

{

Console.Write(i + " ");

}

else

{

Console.Write(i \* (-1) + " ");

}

if (i % 14 == 0) //Added for more perspicuous printing

{

Console.WriteLine();

}

}

}

}

*for (int i = 2; i < 1002; i++) { Console.WriteLine(Math.Pow(-1, i)\*i); }*

**Primitive Data Types and Variables**

//Problem 3. **Variable in Hexadecimal Format**

//Declare an integer variable and assign it with the value 254 in hexadecimal format (0x##).

//Use Windows Calculator to find its hexadecimal representation.

//Print the variable and ensure that the result is 254.

using System;

class HexVariable

{

static void Main()

{

int var = 0xFE;

Console.WriteLine(var);

}

}

//Problem 4. **Unicode Character**

//Declare a character variable and assign it with the symbol that has Unicode code 42 (decimal) using the \u00XX syntax, and then print it.

//Hint: first, use the Windows Calculator to find the hexadecimal representation of 42. The output should be \*.

using System;

class UnicodeCharacter

{

static void Main()

{

char symbol = '\u002A';

Console.WriteLine(symbol);

}

}

//Problem 5. **Boolean Variable**

//Declare a Boolean variable called isFemale and assign an appropriate value corresponding to your gender.

//Print it on the console.

using System;

class MyGender

{

static void Main()

{

Boolean isFemale = true;

Console.WriteLine("\"I'm female\" is {0}.", isFemale);

}

}

//Problem 6. **Strings and Objects**

//Declare two string variables and assign them with Hello and World.

//Declare an object variable and assign it with the concatenation of the first two variables (mind adding an interval between).

//Declare a third string variable and initialize it with the value of the object variable (you should perform type casting).

using System;

class StringAndObject

{

static void Main()

{

string string1 = "Hello";

string string2 = "World";

object concat = string1 + " " + string2;

string string3 = (string)concat; //string string3 = concat.ToString();

Console.WriteLine("The first string is: {0}\nThe second string is: {1}\nThe concatenation is: {2} \nThe third string is: {3}", string1, string2, concat, string3);

}

}

//Problem 7. **Quotes in Strings**

//Declare two string variables and assign them with following value: The "use" of quotations causes difficulties.

//Do the above in two different ways - with and without using quoted strings.

//Print the variables to ensure that their value was correctly defined.

using System;

class StringsWithWithoutQuoted

{

static void Main()

{

string withQuoted = @"The ""use"" of quotations causes difficulties.";

string withoutQuoted = "The \"use\" of quotations causes difficulties.";

Console.WriteLine("With Quoted strings: {0}\nWhithout Quoted strings: {1}", withQuoted, withoutQuoted);

}

}

//Problem 8**. Isosceles Triangle**

//Write a program that prints an isosceles triangle of 9 copyright symbols ©, something like this:

// ©

// © ©

// © ©

//© © © ©

//Note: The © symbol may be displayed incorrectly at the console so you may need to change the console character encoding to UTF-8

//and assign a Unicode-friendly font in the console.

//Note: Under old versions of Windows the © symbol may still be displayed incorrectly,

//regardless of how much effort you put to fix it.

using System;

class TriangleWith9CopyrightSymbols

{

static void Main()

{

Console.OutputEncoding = System.Text.Encoding.UTF8;

char symbol = (char)0xA9;

//// other way

//char symbol = '\u00A9';

Console.WriteLine(" " + symbol + " ");

Console.WriteLine(" " + symbol + " " + symbol + " ");

Console.WriteLine(" " + symbol + " " + symbol + " ");

Console.WriteLine(symbol + " " + symbol + " " + symbol + " " + symbol);

}

}

//Problem 12. **Null Values Arithmetic**

//Create a program that assigns null values to an integer and to a double variable.

//Try to print these variables at the console.

//Try to add some number or the null literal to these variables and print the result.

using System;

class AssignNullValues

{

static void Main()

{

int? intVar = null;

double? doubleVar = null;

Console.WriteLine("The values of int is: {0}\nThe values of double is: {1}", intVar, doubleVar);

intVar += 5;

Console.WriteLine("Add some number = 5: {0}", intVar);

doubleVar += null;

Console.WriteLine("Add the null literal: {0}", doubleVar);

// other

Console.WriteLine("If we use GetValueOrDefault() the result is:");

Console.WriteLine(intVar.GetValueOrDefault() + 6);

Console.WriteLine(doubleVar.GetValueOrDefault() + 7);

}

}

/\*Problem 13.\* **Comparing Floats**

Write a program that safely compares floating-point numbers (double) with precision eps = 0.000001.

Note: Two floating-point numbers a and b cannot be compared directly by a == b

because of the nature of the floating-point arithmetic.

Therefore, we assume two numbers are equal if they are more closely to each other than a fixed constant eps.

Examples:

Number a Number b Equal (with precision eps=0.000001) Explanation

5.3 6.01 false The difference of 0.71 is too big (> eps)

5.00000001 5.00000003 true The difference 0.00000002 < eps

5.00000005 5.00000001 true The difference 0.00000004 < eps

-0.0000007 0.00000007 true The difference 0.00000077 < eps

-4.999999 -4.999998 false Border case. The difference 0.000001 == eps.

We consider the numbers are different.

4.999999 4.999998 false Border case. The difference 0.000001 == eps.

We consider the numbers are different.

\*/

using System;

using System.Threading;

using System.Globalization;

class ComparingFloats

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = CultureInfo.InvariantCulture;

Console.Write("Enter first number a = ");

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

Console.Write("Enter second number b = ");

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

double eps = 0.000001;

double difference = Math.Abs(a - b);

// first way

if (difference < eps)

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

// other way bool difference = difference < eps;

}

}

/\*Problem 14.\* **Print the ASCII Table**

\* Find online more information about ASCII (American Standard Code for Information Interchange)

\* and write a program that prints the entire ASCII table of characters on the console (characters from 0 to 255).

\* Note: Some characters have a special purpose and will not be displayed as expected.

You may skip them or display them differently.

\* Note: You may need to use for-loops (learn in Internet how). \*/

using System;

using System.Text;

class PrintTheASCIITable

{

static void Main()

{

Console.BufferHeight = 256;

Console.OutputEncoding = Encoding.Unicode;

for (int i = 0; i <= 255; i++)

{

Console.WriteLine("{0} => {1}", i, (char)i); // Convert.ToChar(i);

}

}

}

**Operators and Expressions**

/\*Problem 1. **Odd or Even Integers**

Write an expression that checks if given integer is odd or even.

Examples:

n Odd?

3 true

2 false

-2 false

-1 true

0 false \*/

using System;

class OddOrEvenInteger

{

static void Main()

{

Console.WriteLine("Insert a number of type Integer: ");

int intNumber = int.Parse(Console.ReadLine());/\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing.\*/

Console.WriteLine((intNumber % 2 == 0) ? "The number {0} is even." : "The number {0} is odd.", intNumber);

Console.Write("The statement 'The number {0} is odd' is: ", intNumber);

bool odd = (intNumber % 2 != 0);

Console.WriteLine("{0} \n", odd);

}

}

/\*Problem 2. **Gravitation on the Moon**

The gravitational field of the Moon is approximately 17% of that on the Earth.

Write a program that calculates the weight of a man on the moon by a given weight on the Earth.

Examples:

weight weight on the Moon

86 14.62

74.6 12.682

53.7 9.129 \*/

using System;

using System.Threading; // line 11 and 12 are necessary for the line 18

using System.Globalization;

class WeightOnTheMoon

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = new CultureInfo("en-US");

/\* changes the culture of the program in American standard so the decimal point character is . \*/

Console.WriteLine("Insert the weight of a man on the Earth \n (for real value please use the decimal point character '.'): ");

double weightOnTheEarth = double.Parse(Console.ReadLine().Replace(",", "."));

/\* parsing and replacing the character ',' with '.' for ensuring the correct result even with introduced ","-format \*/

double weightOnTheMoon = weightOnTheEarth \* 0.17;

Console.WriteLine("The weight of the man on the moon will be: {0} \n", weightOnTheMoon);

}

}

/\* Problem 3. **Divide by 7 and 5**

Write a Boolean expression that checks for given integer if it can be divided (without remainder) by 7 and 5 at the same time.

Examples:

n Divided by 7 and 5?

3 false

0 true

5 false

7 false

35 true

140 true \*/

using System;

class DividedBy7And5

{

static void Main()

{

Console.WriteLine("Insert an integer number: ");

int intNumber = int.Parse(Console.ReadLine()); // parsing

bool devided = (intNumber % 35 == 0);

// another way: bool devided = ((intNumber % 7 == 0) && (intNumber % 5 == 0));

Console.WriteLine("'The number can be divided by 7 and 5' = " + devided);

}

}

/\* Problem 5. **Third Digit is 7?**

Write an expression that checks for given integer if its third digit from right-to-left is 7.

Examples:

n Third digit 7?

5 false

701 true

9703 true

877 false

777877 false

9999799 true \*/

using System;

class WhetherThirdDigitIs7

{

static void Main()

{

Console.WriteLine("Insert a number of type Integer: ");

int intNumber = int.Parse(Console.ReadLine());

/\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

if (intNumber < 100)

{

Console.WriteLine("'The third digit from right-to-left is 7' = false");

}

else

{

int dividedBy100 = intNumber / 100;

int remainderBy10 = dividedBy100 % 10;

Console.WriteLine((remainderBy10 == 7) ? "'The third digit from right-to-left is 7' = true" : "'The third digit from right-to-left is 7' = false");

}

}

}

/\* Problem 6. **Four-Digit Number**

Write a program that takes as input a four-digit number in format abcd (e.g. 2011) and performs the following:

Calculates the sum of the digits (in our example 2 + 0 + 1 + 1 = 4).

Prints on the console the number in reversed order: dcba (in our example 1102).

Puts the last digit in the first position: dabc (in our example 1201).

Exchanges the second and the third digits: acbd (in our example 2101).

The number has always exactly 4 digits and cannot start with 0.

Examples:

n sum of digits reversed last digit in front second and third digits exchanged

2011 4 1102 1201 2101

3333 12 3333 3333 3333

9876 30 6789 6987 9786 \*/

using System;

class FourDigitNumber

{

static void Main()

{

Console.WriteLine("Insert a four-digit integer number: ");

int number = int.Parse(Console.ReadLine()); //parsing

while ((number < 1000) || (number > 9999)) //if we want to check

{

Console.WriteLine("Incorrect number format. Please insert a four-digit integer number: ");

number = int.Parse(Console.ReadLine());

}

int digit4 = number % 10;

int digit3 = number / 10 % 10;

int digit2 = number / 100 % 10;

int digit1 = number / 1000;

Console.WriteLine("The sum of the digits is: " + (digit1 + digit2 + digit3 + digit4));

Console.WriteLine("The number in reversed order: {3}{2}{1}{0}", digit1, digit2, digit3, digit4);

Console.WriteLine("Putting the last digit in front: {3}{0}{1}{2}", digit1, digit2, digit3, digit4);

Console.WriteLine("Exchanging the second and the third digits: {0}{2}{1}{3}", digit1, digit2, digit3, digit4);

}

}

//Other way

Console.Write("Enter four digit : ");

string number = Console.ReadLine();

int sum = 0;

string conversed = "";

string dabc = "";

string acbd = "";

for (int i = 0; i < 4; i++)

{

string oneDigit = Convert.ToString(number[i]);

sum = sum + Convert.ToInt32(oneDigit);

conversed += number[-(i - 3)];

dabc += (i == 0 ? number[3] : number[i - 1]);

acbd += (((i == 1) || (i == 2)) ? number[-(i - 3)] : number[i]);

}

Console.WriteLine("Sum is : " + sum);

Console.WriteLine("Conversed string : " + conversed);

Console.WriteLine("Changed as DABC : " + dabc);

Console.WriteLine("Changed as ACBD : " + acbd);

/\* Problem 7. **Point in a Circle**

Write an expression that checks if given point (x, y) is inside a circle K({0, 0}, 2).

Examples:

x y inside

0 1 true

-2 0 true

-1 2 false

1.5 -1 true

-1.5 -1.5 false

100 -30 false

0 0 true

0.2 -0.8 true

0.9 -1.93 false

1 1.655 true \*/

using System;

using System.Threading; // line 17 and 18 are necessary for the line 24

using System.Globalization;

class PointInACircle

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = new CultureInfo("en-US");

/\* changes the culture of the program in American standard so the decimal point character is . \*/

Console.WriteLine("Insert the coordinates of Point (x, y) \n using the decimal point character '.' for real values: ");

Console.Write("x= ");

double x = double.Parse(Console.ReadLine().Replace(",", ".")); /\* parsing and replacing the character ',' with '.' for ensuring the correct result even with introduced ","-format \*/

Console.Write("y= ");

double y = double.Parse(Console.ReadLine().Replace(",", ".")); //parsing and replacing

Console.WriteLine((x \* x + y \* y <= 4) ? "inside = true" : "inside = false");

// using Pythagorean theorem

}

}

/\* Problem 8. **Prime Number Check**

Write an expression that checks if given positive integer number n (n <= 100) is prime

(i.e. it is divisible without remainder only to itself and 1).

Note: You should check if the number is positive

Examples:

n Prime?

1 false

2 true

3 true

4 false

9 false

97 true

51 false

-3 false

0 false \*/

using System;

class PrimeNumberCheck

{

static void Main()

{

Console.WriteLine("Insert a positive integer number n: n <= 100: ");

int posIntNumber = int.Parse(Console.ReadLine());

/\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

while ((posIntNumber < 0) || (posIntNumber > 100)) // input check n <= 100

{

Console.WriteLine("Incorrect number format. Please insert a positive integer number n:

n <= 100: ");

posIntNumber = int.Parse(Console.ReadLine());

}

bool boolPrime = true;

if ((posIntNumber == 2) || (posIntNumber == 5))

{

boolPrime = true;

}

else

{

if ((posIntNumber <= 1) || (posIntNumber % 2 == 0) || (posIntNumber % 5 == 0))

{

boolPrime = false;

}

else

{

int maxDivider = (int)Math.Sqrt(posIntNumber);

int count = 0;

for (int i = 3; i <= maxDivider; i++)

{

if (posIntNumber % i == 0)

{

count++;

}

}

if (count > 0)

{

boolPrime = false;

}

}

}

Console.WriteLine("'The given positive integer number {0} is prime' = {1} \n",

posIntNumber, boolPrime);

}

}

/\* Problem 9. **Trapezoids**

Write an expression that calculates trapezoid's area by given sides a and b and height h.

Examples:

a b h area

5 7 12 72

2 1 33 49.5

8.5 4.3 2.7 17.28

100 200 300 45000

0.222 0.333 0.555 0.1540125

\*/

using System;

using System.Threading; // line 14 and 15 are necessary for the line 21

using System.Globalization;

class TrapezoidsArea

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = new CultureInfo("en-US");

/\* changes the culture of the program in American standard so the decimal point character is . \*/

Console.WriteLine("Insert the trapezoid’s side a \n (for real value please use the decimal point character '.'): ");

double a = double.Parse(Console.ReadLine().Replace(",", "."));

/\* parsing and replacing the character ',' with '.' for ensuring the correct result even with introduced ","-format \*/

Console.WriteLine("Insert the trapezoid’s side b \n (for real value please use the decimal point character '.'): ");

double b = double.Parse(Console.ReadLine().Replace(",", ".")); // parsing and replacing

Console.WriteLine("Insert the trapezoid’s height h \n (for real value please use the decimal point character '.'): ");

double h = double.Parse(Console.ReadLine().Replace(",", ".")); // parsing and replacing

Console.WriteLine("The trapezoid's area is: " + ((a + b) \* h / 2));

}

}

/\* Problem 10. **Point Inside a Circle & Outside of a Rectangle**

Write an expression that checks for given point (x, y) if it is

within the circle K({1, 1}, 1.5) and out of the rectangle R(top=1, left=-1, width=6, height=2).

Examples:

x y inside K & outside of R

1 2 yes

2.5 2 no

0 1 no

2.5 1 no

2 0 no

4 0 no

2.5 1.5 no

2 1.5 yes

1 2.5 yes

-100 -100 no

\*/

using System;

using System.Threading; // line 19 and 20 are necessary for the line 26

using System.Globalization;

class PointInCircleOutRectangle

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = new CultureInfo("en-US");

/\* changes the culture of the program in American standard so the decimal point character is . \*/

double x1 = 1; // circle (01{x1,y1}, r)

double y1 = 1;

double r = 1.5;

Console.WriteLine("Insert the coordinates of Point (x, y) \n using the decimal point character '.' for real values: ");

Console.Write("x= ");

double x2 = double.Parse(Console.ReadLine().Replace(",", "."));

/\* parsing and replacing the character ',' with '.'for ensuring the correct result even with introduced ","-format \*/

Console.Write("y= ");

double y2 = double.Parse(Console.ReadLine().Replace(",", ".")); // parcing and replacing

bool insideCircle = (x2 - x1) \* (x2 - x1) + (y2 - y1) \* (y2 - y1) <= r \* r;

// using Pythagorean theorem

double top = 1; // rectangle's parameters

double left = -1;

double width = 6;

double height = 2;

bool insideRectangle = (x2 >= left) && (x2 <= (left + width)) &&

(y2 >= (top - height)) && (y2 <= top);

string insideCOutsideR = (insideCircle && !insideRectangle) ? "yes" : "no";

Console.WriteLine("'The Point ({0}, {1}) is inside the circle and outside of rectangle' = {2}",

x2, y2, insideCOutsideR);

}

}

/\* Problem 11. Bitwise: **Extract Bit #3**

Using bitwise operators, write an expression for finding the value of the bit #3 of a given unsigned integer.

The bits are counted from right to left, starting from bit #0.

The result of the expression should be either 1 or 0.

Examples:

n binary representation bit #3

5 00000000 00000101 0

0 00000000 00000000 0

15 00000000 00001111 1

5343 00010100 11011111 1

62241 11110011 00100001 0 \*/

using System;

class BitwiseExtractBit3

{

static void Main()

{

Console.WriteLine("Insert an unsigned integer number: ");

uint number = uint.Parse(Console.ReadLine());

/\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("The binary representation of the {0} is: {1}", number,

Convert.ToString(number, 2).PadLeft(16, '0'));

uint numberRight3 = number >> 3;

uint bit = numberRight3 & 1;

Console.WriteLine("The bit #3 is: " + bit);

}

}

/\* Problem 12. **Extract Bit from Integer**

Write an expression that extracts from given integer n the value of given bit at index p.

Examples:

n binary representation p bit @ p

5 00000000 00000101 2 1

0 00000000 00000000 9 0

15 00000000 00001111 1 1

5343 00010100 11011111 7 1

62241 11110011 00100001 11 0 \*/

using System;

class ExtractPBitFromIntegerN

{

static void Main()

{

Console.WriteLine("Insert an integer number: ");

int number = int.Parse(Console.ReadLine());

/\* Read the input as a string (using ReadLine()) and convert this string to a number.

The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("Insert an index (position) of the bit, which value you want to extract: ");

int position = int.Parse(Console.ReadLine());

while (position < 0)

{

Console.WriteLine("Incorrect position format. Please insert an integer index: position >= 0: ");

position = int.Parse(Console.ReadLine());

}

int numberRightP = number >> position;

int bit = numberRightP & 1;

Console.WriteLine("The bit #{0} is: {1}", position, bit);

}

}

/\* Problem 13. **Check a Bit at Given Position**

Write a Boolean expression that returns if the bit at position p (counting from 0, starting from the right)

in given integer number n, has value of 1.

Examples:

n binary representation of n p bit @ p == 1

5 00000000 00000101 2 true

0 00000000 00000000 9 false

15 00000000 00001111 1 true

5343 00010100 11011111 7 true

62241 11110011 00100001 11 false \*/

using System;

class CheckIfPBitIs1

{

static void Main()

{

Console.WriteLine("Insert an integer number: ");

int number = int.Parse(Console.ReadLine());

/\* Read the input as a string (using ReadLine()) and convert this string to a number.

The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("Insert an index (position) of the bit, which value you want to check: ");

int position = int.Parse(Console.ReadLine());

while (position < 0)

{

Console.WriteLine("Incorrect position format. Please insert an integer index: position >= 0: ");

position = int.Parse(Console.ReadLine());

}

int numberRightP = number >> position;

int bit = numberRightP & 1;

Console.WriteLine((bit == 1) ? "true" : "false");

}

}

/\* Problem 14. **Modify a Bit at Given Position**

We are given an integer number n, a bit value v (v=0 or 1) and a position p.

Write a sequence of operators (a few lines of C# code) that modifies n to hold the value v at the position p from the binary representation of n while preserving all other bits in n.

Examples:

n binary representation of n p v binary result result

5 00000000 00000101 2 0 00000000 00000001 1

0 00000000 00000000 9 1 00000010 00000000 512

15 00000000 00001111 1 1 00000000 00001111 15

5343 00010100 11011111 7 0 00010100 01011111 5215

62241 11110011 00100001 11 0 11110011 00100001 62241 \*/

using System;

class ModifyPBitToValue

{

static void Main()

{

Console.WriteLine("Insert an integer number: ");

int number = int.Parse(Console.ReadLine()); /\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("Insert an index (position) of the bit, which value you want to modify: ");

int position = int.Parse(Console.ReadLine());

while (position < 0)

{

Console.WriteLine("Incorrect position format. Please insert an integer index: position >= 0: ");

position = int.Parse(Console.ReadLine());

}

Console.WriteLine("Insert a value (0 or 1): ");

int value = int.Parse(Console.ReadLine());

while ((value != 0) && (value != 1))

{

Console.WriteLine("Incorrect value. Please insert a value 0 or 1: ");

value = int.Parse(Console.ReadLine());

}

if (value == 0)

{

int mask = ~(1 << position);

int result = number & mask;

Console.WriteLine("The binary result is: " + Convert.ToString(result, 2).PadLeft(16, '0')); // print the binary number

Console.WriteLine("The result is: " + result + "\n");

}

else

{

int mask = 1 << position;

int result = number | mask;

Console.WriteLine("The binary result is: " + Convert.ToString(result, 2).PadLeft(16, '0'));

Console.WriteLine("The result is: " + result + "\n");

}

}

}

/\* Problem 15.\* **Bits Exchange**

Write a program that exchanges bits 3, 4 and 5 with bits 24, 25 and 26 of given 32-bit unsigned integer.

Examples:

n binary representation of n binary result result

1140867093 01000100 00000000 01000000 00010101 01000010 00000000 01000000 00100101 1107312677

255406592 00001111 00111001 00110010 00000000 00001000 00111001 00110010 00111000 137966136

4294901775 11111111 11111111 00000000 00001111 11111001 11111111 00000000 00111111 4194238527

5351 00000000 00000000 00010100 11100111 00000100 00000000 00010100 11000111 67114183

2369124121 10001101 00110101 11110111 00011001 10001011 00110101 11110111 00101001 2335569705

\*/

using System;

class BitsExchange

{

static void Main()

{

Console.WriteLine("Insert an unsighned integer number: ");

uint number = uint.Parse(Console.ReadLine()); /\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("Binary representation of the number before the exchange: \n" + Convert.ToString(number, 2).PadLeft(32, '0') + "\n");

uint[] pArray = new uint[3];

uint[] qArray = new uint[3];

uint result;

for (int position = 3; position < 6; position++) // get the bits at position 3, 4 and 5 and put them in pArray

{

uint numberRightP = number >> position;

pArray[position - 3] = numberRightP & 1;

}

for (int position = 24; position < 27; position++) // get the bits at position 24, 25 and 26 and put them in qArray

{

uint numberRightP = number >> position;

qArray[position - 24] = numberRightP & 1;

}

for (int position = 3; position < 6; position++) // exchange the bits at position 3, 4 and 5

{

if (qArray[position - 3] == 0) // to 0

{

uint mask = (uint)(~(1 << position));

result = number & mask;

}

else // to 1

{

uint mask = (uint)(1 << position);

result = number | mask;

}

number = result;

}

for (int position = 24; position < 27; position++) // exchange the bits at position 24, 25 and 26

{

if (pArray[position - 24] == 0) // to 0

{

uint mask = (uint)(~(1 << position));

result = number & mask;

}

else // to 1

{

uint mask = (uint)(1 << position);

result = number | mask;

}

number = result;

}

Console.WriteLine("The binary result after the axchange is: \n" + Convert.ToString(number, 2).PadLeft(32, '0') + "\n");

Console.WriteLine("The result is: " + number + "\n");

}

}

/\* Problem 16.\*\* **Bit Exchange (Advanced)**

Write a program that exchanges bits {p, p+1, …, p+k-1} with bits {q, q+1, …, q+k-1} of a given 32-bit unsigned integer.

The first and the second sequence of bits may not overlap.

Examples:

n p q k binary representation of n binary result result

1140867093 3 24 3 01000100 00000000 01000000 00010101 01000010 00000000 01000000 00100101 1107312677

4294901775 24 3 3 11111111 11111111 00000000 00001111 11111001 11111111 00000000 00111111 4194238527

2369124121 2 22 10 10001101 00110101 11110111 00011001 01110001 10110101 11111000 11010001 1907751121

987654321 2 8 11 00111010 11011110 01101000 10110001 - overlapping

123456789 26 0 7 00000111 01011011 11001101 00010101 - out of range

33333333333 -1 0 33 00000111 11000010 11010010 01001101 01010101 - out of range

\*/

using System;

class BitsExchange

{

static void Main()

{

Console.WriteLine("Insert an unsighned integer number: ");

ulong number = ulong.Parse(Console.ReadLine()); /\* Read the input as a string (using ReadLine()) and convert this string to a number.

\* The operation of converting a string into another type is called parsing. \*/

Console.WriteLine("Binary representation of the number before the exchange: \n" + Convert.ToString((uint)number, 2).PadLeft(32, '0') + "\n");

Console.Write("Enter p= ");

int p = int.Parse(Console.ReadLine());

Console.Write("Enter q= ");

int q = int.Parse(Console.ReadLine());

Console.Write("Enter k= ");

int k = int.Parse(Console.ReadLine());

if (Math.Max(p, q) + k > 32)

{

Console.WriteLine("out of range");

}

else if (Math.Min(p, q) + k > Math.Max(p, q))

{

Console.WriteLine("overlapping");

}

else

{

uint[] pArray = new uint[k];

uint[] qArray = new uint[k];

uint result;

for (int position = p; position < p + k; position++)//get the bits at position p, p+1, ..., p+k-1 and put them in pArray

{

uint numberRightP = ((uint)number) >> position;

pArray[position - p] = numberRightP & 1;

}

for (int position = q; position < q + k; position++)//get the bits at position q, q+1, ..., q+k-1 and put them in qArray

{

uint numberRightP = ((uint)number) >> position;

qArray[position - q] = numberRightP & 1;

}

for (int position = p; position < p + k; position++) // exchange the bits at position p, p+1, ..., p+k-1

{

if (qArray[position - p] == 0) // to 0

{

uint mask = (uint)(~(1 << position));

result = ((uint)number) & mask;

}

else // to 1

{

uint mask = (uint)(1 << position);

result = ((uint)number) | mask;

}

number = result;

}

for (int position = q; position < q + k; position++) // exchange the bits at position q, q+1, ..., q+k-1

{

if (pArray[position - q] == 0) // to 0

{

uint mask = (uint)(~(1 << position));

result = ((uint)number) & mask;

}

else // to 1

{

uint mask = (uint)(1 << position);

result = ((uint)number) | mask;

}

number = result;

}

Console.WriteLine("The binary result after the axchange is: \n" + Convert.ToString(((uint)number), 2).PadLeft(32, '0') + "\n");

Console.WriteLine("The result is: " + number + "\n");

}

}

}

…………………………………………………………………………………………………………..

using System;

using System.Text;

class StringBuilderDemo

{

public static string ReverseIt(string str)

{

StringBuilder sb = new StringBuilder();

// var sb = new StringBuilder(str.Length);

for (int i = str.Length - 1; i >= 0; i--)

{

sb.Append(str[i]);

}

return sb.ToString();

}

public static string ExtractCapitals(string str)

{

StringBuilder result = new StringBuilder();

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

{

char ch = str[i];

if (Char.IsUpper(ch))

//if (s[i] >= 'A' && s[i] <= 'Z')

{

result.Append(ch);

}

}

return result.ToString();

}

public static string DupChar(char ch, int count)

{

StringBuilder result = new StringBuilder(count);

for (int i = 0; i < count; i++)

{

result.Append(ch);

}

return result.ToString();

}

static void Main()

{

string str = "Telerik Academy";

string reversed = ReverseIt(str);

Console.WriteLine(reversed); // ymedacA kireleT

string capitals = ExtractCapitals(str);

Console.WriteLine(capitals); // TA

Console.WriteLine(DupChar('a', 10)); // aaaaaaaaaa

}

}

…………………………………………………………………………………………………………………………………………………………………………………………………………………………..

**Homework: Arrays**

/\* Problem 1. Allocate array

Write a program that allocates array of 20 integers and initializes each element by its index multiplied by 5.

Print the obtained array on the console.

\*/

using System;

class AllocateArray

{

static void Main()

{

var intArray = new int[20]; //int[] intArray = new int[20];

for (int index = 0; index < intArray.Length; index++)

{

intArray[index] = 5 \* index;

}

for (int index = 0; index < intArray.Length; index++)

{

Console.WriteLine("element[{0}] = {1}", index, intArray[index]);

}

}

}

/\* Problem 2. Compare arrays

Write a program that reads two integer arrays from the console and compares them element by element.

\*/

using System;

class CompareArrays

{

static void Main()

{

Console.WriteLine("Please enter the length of the first array:");

int length1 = int.Parse(Console.ReadLine()); // read from the console the length of the array1

int[] array1 = new int[length1]; // create the array1 of given size

Console.WriteLine("Please enter the elements of the first array: ");

for (int i = 0; i < array1.Length; i++)

{

Console.Write("element[{0}] = ", i);

array1[i] = int.Parse(Console.ReadLine()); // read the elements of the array1 in a for loop

Console.WriteLine();

}

Console.WriteLine("Please enter the length of the second array:");

int length2 = int.Parse(Console.ReadLine()); // read from the console the length of the array2

int[] array2 = new int[length2]; // create the array2 of given size

Console.WriteLine("Please enter the elements of the first array: ");

for (int i = 0; i < array2.Length; i++)

{

Console.Write("element[{0}] = ", i);

array2[i] = int.Parse(Console.ReadLine()); // read the elements of the array2 in a for loop

Console.WriteLine();

}

if (length1 != length2)

{

Console.WriteLine("The arrays have a different value for the length.");

}

else

{

bool equal = true;

for (int i = 0; i < length1; i++)

{

if (array1[i] != array2[i])

{

equal = false;

break;

}

}

if (equal)

{

Console.WriteLine("The elements of the arrays are equal.");

}

else

{

Console.WriteLine("The elements of the arrays are not equal.");

}

}

}

}

/\* Problem 3. Compare char arrays

Write a program that compares two char arrays lexicographically (letter by letter).

\*/

/\*Notes: In lexicographic order the elements are compared one by one starting from the very left. If the elements are not the same, the array, whose element is smaller (comes earlier in the alphabet), comes first. If the elements are equal, the next character is compared. If the end of one of the arrays is reached, without finding different elements, the shorter array is the smaller (comes earlier lexicographically). If all elements are equal, the arrays are equal.

\*/

using System;

class CompareCharArrays

{

static void Main()

{

Console.WriteLine("Please enter the first char array (on a single line without spaces): ");

string firstChars = Console.ReadLine();

char[] firstArray = firstChars.ToCharArray(); // convert string to char array

int firstArrayLength = firstArray.Length; // gets length of the first array

Console.WriteLine("Please enter the second char array (on a single line without spaces): ");

string secondChars = Console.ReadLine();

char[] secondArray = secondChars.ToCharArray();

int secondArrayLength = secondArray.Length;

int minLength = Math.Min(secondArrayLength, firstArrayLength); // gets min length of the two arrays

bool equalChars = true;

for (int i = 0; i < minLength; i++)

{

if (firstArray[i] < secondArray[i])

{

Console.WriteLine("The first array comes earlier lexicographically.");

equalChars = false;

break;

}

else if (firstArray[i] > secondArray[i])

{

Console.WriteLine("The second array comes earlier lexicographically.");

equalChars = false;

break;

}

}

if (equalChars) // if no differences are found

{

if (firstArrayLength < secondArrayLength)

{

Console.WriteLine("The first array (the shorter) comes earlier lexicographically.");

}

else if (firstArrayLength > secondArrayLength)

{

Console.WriteLine("The second array (the shorter) comes earlier lexicographically.");

}

else

{

Console.WriteLine("The arrays are equal. (All elements are equal.)");

}

}

}

}

/\* Problem 4. Maximal sequence

Write a program that finds the maximal sequence of equal elements in an array.

Example:

input result

2, 1, 1, 2, 3, 3, 2, 2, 2, 1 2, 2, 2

\*/

// Note: The equal elements are consecutively placed.

using System;

class MaximalSequence

{

static void Main()

{

//Input

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

//Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

//Fill an integer array from string array

int[] arrayNums = new int[inputElements.Length];

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

{

arrayNums[i] = int.Parse(inputElements[i]);

}

int currentLength = 1;

int maxLength = 0;

int elementBestLength = 0;

for (int i = 0; i < arrayNums.Length - 1; i++)

{

if (arrayNums[i] == arrayNums[i + 1])

{

currentLength++;

}

else

{

if (currentLength > maxLength)

{

maxLength = currentLength;

elementBestLength = arrayNums[i];

}

currentLength = 1;

}

}

// in case when the last sequence is the maximal sequence of equal elements

if (currentLength > maxLength)

{

maxLength = currentLength;

elementBestLength = arrayNums[arrayNums.Length - 1];

}

for (int i = 0; i < maxLength; i++)

{

Console.Write(i != maxLength - 1 ? elementBestLength + ", " : elementBestLength + "\n");

}

}

}

/\* Problem 5. Maximal increasing sequence

Write a program that finds the maximal increasing sequence in an array.

Example:

input result

3, 2, 3, 4, 2, 2, 4 2, 3, 4

\*/

// Note: The increasing elements are consecutively placed.

using System;

class MaximalIncreasingSequence

{

static void Main()

{

//Input

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

//Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

//Fill an integer array from string array

int[] elementsArray = new int[inputElements.Length];

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

{

elementsArray[i] = int.Parse(inputElements[i]);

}

int currentLength = 1;

int maxLength = 0;

int endIndexBestLength = 0;

for (int i = 0; i < elementsArray.Length - 1; i++)

{

if (elementsArray[i] < elementsArray[i + 1])

{

currentLength++;

}

else

{

if (currentLength > maxLength)

{

maxLength = currentLength;

endIndexBestLength = i;

}

currentLength = 1;

}

}

// in case when the last sequence is the maximal sequence of equal elements

if (currentLength > maxLength)

{

maxLength = currentLength;

endIndexBestLength = elementsArray.Length - 1;

}

for (int i = endIndexBestLength - maxLength + 1; i <= endIndexBestLength; i++)

{

Console.Write(i != endIndexBestLength ? elementsArray[i] + ", " : elementsArray[i] + "\n");

}

}

}

/\* Problem 6. Maximal K sum

Write a program that reads two integer numbers N and K and an array of N elements from the console.

Find in the array those K elements that have maximal sum.

\*/

// Interpretation: It is not necessary the elements to be consecutively placed.

using System;

class MaximalKSum

{

static void Main()

{

Console.Write("Enter an integer number N: ");

int n;

while (!int.TryParse(Console.ReadLine(), out n)) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid number.");

Console.Write("Please enter an integer number for N: ");

}

Console.Write("Enter an integer number K: ");

int k;

while (!int.TryParse(Console.ReadLine(), out k) || k > n) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid number.");

Console.Write("Please enter an integer number for K: ");

}

int[] array = new int[n];

Console.WriteLine("Enter {0} numbers to array:", n);

for (int i = 0; i < n; i++)

{

array[i] = int.Parse(Console.ReadLine());

}

Array.Sort(array);

Array.Reverse(array);

Console.WriteLine("The {0} elements that have maximal sum are: ", k);

for (int i = 0; i < k; i++)

{

Console.Write(i != k - 1 ? array[i] + ", " : array[i] + "\n");

}

}

}

/\* Problem 6. Maximal K sum - Consecutive

Write a program that reads two integer numbers N and K and an array of N elements from the console.

Find in the array those K elements that have maximal sum.

\*/

// Interpretation: The elements are consecutively placed.

using System;

class MaximalKSum

{

static void Main()

{

Console.Write("Enter an integer number N: ");

int n;

while (!int.TryParse(Console.ReadLine(), out n)) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid number.");

Console.Write("Please enter an integer number for N: ");

}

Console.Write("Enter an integer number K: ");

int k;

while (!int.TryParse(Console.ReadLine(), out k) || k > n) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid number.");

Console.Write("Please enter an integer number for K: ");

}

int[] array = new int[n];

Console.WriteLine("Enter {0} numbers to array:", n);

for (int i = 0; i < n; i++)

{

array[i] = int.Parse(Console.ReadLine());

}

// help values

string maxSeq = "";

int sum = 0;

int maxSum = int.MinValue;

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

{

string currentSeq = "";

// out of the array bounds

if (i + k > array.Length)

{

break;

}

// set current sum

for (int j = i; j < i + k; j++)

{

sum += array[j];

currentSeq = currentSeq + array[j] + ' ';

}

// max sum check

if (sum > maxSum)

{

maxSeq = currentSeq;

maxSum = sum;

}

sum = 0;

}

Console.WriteLine(maxSeq);

Console.WriteLine(maxSum);

}

}

/\* Problem 7. Selection sort

Sorting an array means to arrange its elements in increasing order. Write a program to sort an array.

Use the Selection sort algorithm: Find the smallest element, move it at the first position,

find the smallest from the rest, move it at the second position, etc.

\*/

using System;

class SelectionSort

{

static void Main()

{

Console.WriteLine("Enter a number for N:");

int n = int.Parse(Console.ReadLine());

int[] arrayNums = new int[n];

Console.WriteLine("Enter {0} numbers to array:", n);

for (int i = 0; i < n; i++)

{

arrayNums[i] = int.Parse(Console.ReadLine());

}

// Classical implementation of Selection Sort Algorithm

for (int i = 0; i < arrayNums.Length - 1; i++)

{

// find the min element in the unsorted a[i..n-1]

// assume the min is the first element

int indexMin = i;

// test against elements after i to find the smallest

int j;

for (j = i + 1; j < arrayNums.Length; j++)

{

// if this element is less, then it is the new minimum

if (arrayNums[j] < arrayNums[indexMin])

{

// found new minimum; remember its index

indexMin = j;

}

}

// swaping

if (indexMin != j)

{

int swap = arrayNums[i];

arrayNums[i] = arrayNums[indexMin];

arrayNums[indexMin] = swap;

}

}

Console.WriteLine("After sorting using Selection Sort Algorithm: ");

Console.WriteLine(string.Join(" ", arrayNums) + "\n");

}

}

/\* Problem 8. Maximal sum

Write a program that finds the sequence of maximal sum in given array.

Example:

input result

2, 3, -6, -1, 2, -1, 6, 4, -8, 8 2, -1, 6, 4

Can you do it with only one loop (with single scan through the elements of the array)?

\*/

using System;

class MaximalSum

{

static void Main()

{

//Input

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

//Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

//Fill an integer array from string array

int[] arrayNums = new int[inputElements.Length];

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

{

arrayNums[i] = int.Parse(inputElements[i]);

}

int startIndex = 0, currentSum = 0;

int bestStart = 0, bestEnd = 0, maxSum = int.MinValue;

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

{

if (currentSum <= 0)

{

startIndex = i;

currentSum = 0;

}

currentSum += arrayNums[i];

if (currentSum > maxSum)

{

maxSum = currentSum;

bestStart = startIndex;

bestEnd = i;

}

}

Console.Write("Sequence of maximal sum: ");

for (int i = bestStart; i <= bestEnd; i++)

{

Console.Write(i != bestEnd ? arrayNums[i] + ", " : arrayNums[i] + "\n");

}

Console.WriteLine("Maximal sum: {0}", maxSum);

}

}

/\* Problem 9. Frequent number

Write a program that finds the most frequent number in an array.

Example:

input result

4, 1, 1, 4, 2, 3, 4, 4, 1, 2, 4, 9, 3 4 (5 times)

\*/

using System;

class FrequentNumber

{

static void Main()

{

//Input

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

//Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

//Fill an integer array from string array

int[] arrayNums = new int[inputElements.Length];

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

{

arrayNums[i] = int.Parse(inputElements[i]);

}

// sorting the numbers in ascending order, so the elements with same value will be placed next to each other

Array.Sort(arrayNums);

// find the longest sequence of neighbor equal elements

int currentLength = 1;

int maxLength = 0;

int elementBestLength = 0;

for (int i = 0; i < arrayNums.Length - 1; i++)

{

if (arrayNums[i] == arrayNums[i + 1])

{

currentLength++;

}

else

{

if (currentLength > maxLength)

{

maxLength = currentLength;

elementBestLength = arrayNums[i];

}

currentLength = 1;

}

}

// in case when the last sequence is the maximal sequence of equal elements

if (currentLength > maxLength)

{

maxLength = currentLength;

elementBestLength = arrayNums[arrayNums.Length - 1];

}

// output

Console.WriteLine("{0} ({1} times)", elementBestLength, maxLength);

}

}

/\* Problem 10. Find sum in array

Write a program that finds in given array of integers a sequence of given sum S (if present).

Example:

array S result

4, 3, 1, 4, 2, 5, 8 11 4, 2, 5

\*/

// Note: The elements are consecutively placed.

using System;

class FindSumInArray

{

static void Main()

{

// Input array

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

// Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

// Fill an integer array from string array

int[] arrayNums = new int[inputElements.Length];

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

{

arrayNums[i] = int.Parse(inputElements[i]);

}

// Input S

Console.WriteLine("Enter the sum S = ");

int s = int.Parse(Console.ReadLine());

// The first loop assigns a starting index

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

{

int sum = 0;

/\* The second loop sums the elements from the starting index to the right

\* until this partial sum reaches or is greater than S. \*/

for (int j = i; j < arrayNums.Length; j++)

{

sum = sum + arrayNums[j];

if (sum > s)

{

sum = 0;

break;

}

// If the sum is equal to S, we remember the starting index (from the first loop) and the ending index (from the second loop).

if (sum == s)

{

for (int index = i; index <= j; index++)

{

Console.Write(index != j ? arrayNums[index] + ", " : arrayNums[index] + "\n");

}

}

}

}

}

}

/\* Problem 11. Binary search

Write a program that finds the index of given element in a sorted array of integers by using the Binary search algorithm.

\*/

using System;

class BinarySearch

{

static void Main()

{

// 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

// Input array

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

// Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

// Fill an integer array from string array

int[] array = new int[inputElements.Length];

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

{

array[i] = int.Parse(inputElements[i]);

}

// Input Number

Console.WriteLine("Enter the element, whose index You want to find: ");

int element = int.Parse(Console.ReadLine());

// Binary search works only on sorted arrays.

Array.Sort(array);

int left = 0;

int right = array.Length - 1;

int middle = 0;

int index = -1;

while (right >= left)

{

middle = (left + right) / 2;

if (array[middle] < element)

{

left = middle + 1;

}

else if (array[middle] > element)

{

right = middle - 1;

}

else // array[middle] == element

{

index = middle;

break;

}

}

if (index != -1)

{

Console.WriteLine("The index of the given element is: {0}", index);

}

else

{

Console.WriteLine("There is no number {0} in the array.", element);

}

}

}

/\* Problem 12. Index of letters

Write a program that creates an array containing all letters from the alphabet (A-Z).

Read a word from the console and print the index of each of its letters in the array.

\*/

using System;

class IndexOfLetters

{

static void Main()

{

int[] lettersArray = new int[26];

// creating an array containing all letters from the alphabet (A-Z)

for (int i = 0, next = 0; i < 26; i++, next++)

{

lettersArray[i] = 'A' + next;

}

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

string word = Console.ReadLine();

// printing the index of each of word's letters in the array

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

{

for (int j = 0; j < lettersArray.Length; j++)

{

if (word[i] == lettersArray[j])

{

Console.WriteLine("Letter {0} has index: {1}", word[i], j);

break;

}

}

}

}

}

/\* Problem 14. Quick sort

Write a program that sorts an array of integers using the Quick sort algorithm.

\*/

using System;

using System.Collections.Generic;

class QuickSort

{

static void Main()

{

// 2, 3, 5, 0, 123, 3, 23, 1234, 87

// Input array

Console.WriteLine("Enter an array of integer elements on a single line, separated by commas:");

string input = Console.ReadLine();

// Split the elements into an array

string[] inputElements = input.Split(new char[] { ',', ' ' }, StringSplitOptions.RemoveEmptyEntries);

// Fill an integer array from string array

int[] array = new int[inputElements.Length];

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

{

array[i] = int.Parse(inputElements[i]);

}

// Converting array to list

List<int> arrayList = new List<int>(array);

List<int> sortedArray = QuickSorting(arrayList);

for (int i = 0; i < sortedArray.Count; i++)

{

Console.Write(i != sortedArray.Count - 1 ? sortedArray[i] + ", " : sortedArray[i] + "\n");

}

}

static List<int> QuickSorting(List<int> unsortedList)

{

if (unsortedList.Count <= 1)

{

return unsortedList;

}

int pivot = unsortedList.Count / 2;

int pivotValue = unsortedList[pivot];

unsortedList.RemoveAt(pivot);

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

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

foreach (int element in unsortedList)

{

if (element <= pivotValue)

{

less.Add(element);

}

else

{

greater.Add(element);

}

}

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

result.AddRange(QuickSorting(less));

result.Add(pivotValue);

result.AddRange(QuickSorting(greater));

return result;

}

}

/\* Problem 15. Prime numbers

Write a program that finds all prime numbers in the range [1...10 000 000]. Use the Sieve of Eratosthenes algorithm.

\*/

using System;

class SieveOfEratosthenes

{

static void Main()

{

bool[] primes = new bool[10000000];

// Find all prime numbers in the range [1...10 000 000]

for (int i = 2; i < Math.Sqrt(primes.Length); i++)

{

// Skip those numbers that are not prime

if (primes[i] == false)

{

for (int j = i \* i; j < primes.Length; j += i)

{

primes[j] = true;

}

}

}

// Print all prime numbers in the range [1...10 000 000]

for (int i = 2; i < primes.Length; i++)

{

if (!primes[i])

{

Console.Write(i + " ");

}

}

}

}

/\* Problem 18.\* Remove elements from array

Write a program that reads an array of integers and removes from it a minimal number of elements

in such a way that the remaining array is sorted in increasing order.

Print the remaining sorted array.

Example:

input result

6, 1, 4, 3, 0, 3, 6, 4, 5 1, 3, 3, 4, 5

\*/

// works like that 1 3 4 5

// first way

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class LongestIncreasingSubset

{

static void Main()

{

//Input

string input = Console.ReadLine();

//Split the elements into an array

string[] inputElements = input.Split(new char[] { ' ', ',' }, StringSplitOptions.RemoveEmptyEntries);

//Fill an integer array from string array

int[] elementsArray = new int[inputElements.Length];

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

{

elementsArray[i] = int.Parse(inputElements[i]);

}

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

//Find the longest sequence of identical elements

int startIndex = 0;

int lenghtCount = 1;

int currentCount = 1;

for (int i = 0; i < elementsArray.Length - 1; i++) //could start on index 1 and check current with previous elements

{

if (elementsArray[i] == elementsArray[i + 1])

{

currentCount++;

if (currentCount > lenghtCount)

{

lenghtCount = currentCount;

startIndex = (i + 1) - (lenghtCount - 1);

}

}

else

{

currentCount = 1;

}

}

//Make the sequence currently longest

for (int i = 0; i < lenghtCount; i++)

{

longestSequenceList.Add(elementsArray[startIndex + i]);

}

//Find how many combinations of sequences can be there

long combinations = 1;

for (int i = 0; i < elementsArray.Length; i++) //Instead of Math.Pow

{

combinations \*= 2;

}

for (long combination = 1; combination <= combinations; combination++)

{

//convert current combination number to its binary representation.

//That way we will use the positions with bit "1" with the elements on the same position in the array

string binary = Convert.ToString(combination, 2).PadLeft(elementsArray.Length, '0');

char[] tempArr = binary.ToCharArray();

Array.Reverse(tempArr);

string revBinary = new string(tempArr);

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

int bitsCount = 0;

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

{

if (revBinary[i] == '1')

{

tempList.Add(elementsArray[i]);

bitsCount++;

}

}

if (bitsCount < longestSequenceList.Count) //Speed optimisation

{

continue;

}

int currentLenght = 0;

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

if (tempList.Count > 1) //Avoid cases where the current combination will use only 1 element

{

int biggestNum = tempList[0];

currentLongestSeq.Add(biggestNum);

for (int i = 0; i < tempList.Count - 1; i++)

{

//If the next number in the current combination sequence is bigger add it to final list

if (tempList[i + 1] > biggestNum)

{

biggestNum = tempList[i + 1];

currentLongestSeq.Add(biggestNum);

}

}

currentLenght = currentLongestSeq.Count;

}

if (currentLenght > longestSequenceList.Count)

{

longestSequenceList = currentLongestSeq;

}

}

//Output

for (int i = 0; i < longestSequenceList.Count; i++)

{

Console.Write(longestSequenceList[i] + " ");

}

Console.WriteLine();

}

}

//// second way

//using System;

//using System.Collections.Generic;

//using System.Linq;

//namespace LongestIncreasingSubset

//{

// class SeqList

// {

// static int equalsFirstIndex;

// static int increasingFirstIndex;

// static void Main()

// {

// string input = Console.ReadLine();

// input = input.Trim();

// int[] arr = input.Split(' ', ',').Select(s => int.Parse(s)).ToArray();

// List<int> increasingSeq = getBestSequence(arr, false); //e.g: 1 2 3 4

// List<int> equalSeq = getBestSequence(arr, true); //e.g: 1 1 1 1

// if (increasingSeq.Count < equalSeq.Count ||

// increasingSeq.Count == equalSeq.Count && increasingFirstIndex > equalsFirstIndex)

// {

// increasingSeq = new List<int>(equalSeq);

// }

// foreach (var item in increasingSeq)

// {

// Console.Write(item + " ");

// }

// Console.WriteLine();

// }

// //finds longest non-decreasing subsequence

// static List<int> getBestSequence(int[] arr, bool equal)

// {

// List<int>[] lens = new List<int>[arr.Length];

// int maxIndex = 0;

// for (int currIndex = 0; currIndex < arr.Length; currIndex++)

// {

// bool expression;

// lens[currIndex] = new List<int>();

// lens[currIndex].Add(arr[currIndex]);

// for (int prevIndex = 0; prevIndex < currIndex; prevIndex++)

// {

// expression = arr[prevIndex] < arr[currIndex];

// if (equal == true)

// {

// expression = arr[prevIndex] == arr[currIndex];

// }

// if (expression &&

// lens[currIndex].Count <= lens[prevIndex].Count)

// {

// lens[currIndex] = new List<int>(lens[prevIndex]);

// lens[currIndex].Add(arr[currIndex]);

// if (lens[currIndex].Count > lens[maxIndex].Count)

// {

// maxIndex = currIndex;

// }

// }

// }

// }

// if (equal == false)

// {

// increasingFirstIndex = maxIndex;

// }

// else

// {

// equalsFirstIndex = maxIndex;

// }

// return lens[maxIndex];

// }

// }

//}

# Homework: Multidimensional Arrays

/\* Problem 1. Fill the matrix A

Write a program that fills and prints a matrix of size (n, n) as shown below:

Example for n=4:

1 5 9 13

2 6 10 14

3 7 11 15

4 8 12 16

\*/

using System;

class FillTheMatrixA

{

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows and columns = ");

int rows = int.Parse(Console.ReadLine());

int cols = rows;

// Create (Allocate) the matrix

int[,] matrix = new int[rows, cols];

FillMatrix(matrix);

Console.WriteLine();

PrintingMatrix(matrix);

}

// Set (Fill) the matrix elements

private static void FillMatrix(int[,] matrix)

{

for (int col = 0, index = 1; col < matrix.GetLength(1); col++)

{

for (int row = 0; row < matrix.GetLength(0); row++, index++)

{

matrix[row, col] = index;

}

}

}

// Print the matrix on the console

private static void PrintingMatrix(int[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-2} ", matrix[row, col]); // left aligned

//Console.Write("{0,2} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 1. Fill the matrix B

Write a program that fills and prints a matrix of size (n, n) as shown below:

Example for n=4:

1 8 9 16

2 7 10 15

3 6 11 14

4 5 12 13

\*/

using System;

class FillTheMatrixB

{

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows and columns = ");

int rows = int.Parse(Console.ReadLine());

int cols = rows;

// Create (Allocate) the matrix

int[,] matrix = new int[rows, cols];

FillMatrix(matrix);

Console.WriteLine();

PrintingMatrix(matrix);

}

// Set (Fill) the matrix elements

private static void FillMatrix(int[,] matrix)

{

bool isDirDown = true;

for (int col = 0, row = 0, index = 1; col < matrix.GetLength(1); col++)

{

while (row >= 0 && row < matrix.GetLength(0))

{

matrix[row, col] = index++;

row += isDirDown ? +1 : -1; // Change 'row' value

}

// Change direction and change 'row' value

isDirDown = !isDirDown;

row += isDirDown ? +1 : -1;

}

}

// Print the matrix on the console

private static void PrintingMatrix(int[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-2} ", matrix[row, col]); // left aligned

//Console.Write("{0,2} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 1. Fill the matrix C

Write a program that fills and prints a matrix of size (n, n) as shown below:

Example for n=4:

7 11 14 16

4 8 12 15

2 5 9 13

1 3 6 10

\*/

using System;

class FillTheMatrixC

{

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows and columns = ");

int rows = int.Parse(Console.ReadLine());

int cols = rows;

// Create (Allocate) the matrix

int[,] matrix = new int[rows, cols];

FillMatrix(matrix);

Console.WriteLine();

PrintingMatrix(matrix);

}

// Set (Fill) the matrix elements

private static void FillMatrix(int[,] matrix)

{

for (int row = matrix.GetLength(0) - 1, index = 1; index <= matrix.GetLength(0) \* matrix.GetLength(1); row--)

{

for (int currCol = (row >= 0 ? 0 : -row), currRow = (row >= 0 ? row : 0);

currCol < (matrix.GetLength(1) - (row >= 0 ? row : 0)); )

{

matrix[currRow++, currCol++] = index++;

}

}

}

// Print the matrix on the console

private static void PrintingMatrix(int[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-2} ", matrix[row, col]); // left aligned

//Console.Write("{0,2} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 1. \* Fill the matrix D \*

Write a program that fills and prints a matrix of size (n, n) as shown below:

Example for n=4:

1 12 11 10

2 13 16 9

3 14 15 8

4 5 6 7

\*/

using System;

class FillTheMatrixD

{

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows and columns = ");

int rows = int.Parse(Console.ReadLine());

int cols = rows;

// Create (Allocate) the matrix

int[,] matrix = new int[rows, cols];

FillMatrix(matrix);

Console.WriteLine();

PrintingMatrix(matrix);

}

// Set (Fill) the matrix elements

private static void FillMatrix(int[,] matrix)

{

string direction = "down";

int row = -1, col = 0;

for (int index = 1; index <= matrix.GetLength(0) \* matrix.GetLength(1); index++)

{

if (direction == "down")

{

if (matrix[++row, col] == 0) matrix[row, col] = index;

if (!IsTraversable(matrix, row + 1, col)) direction = "right";

}

else if (direction == "right")

{

if (matrix[row, ++col] == 0) matrix[row, col] = index;

if (!IsTraversable(matrix, row, col + 1)) direction = "up";

}

else if (direction == "up")

{

if (matrix[--row, col] == 0) matrix[row, col] = index;

if (!IsTraversable(matrix, row - 1, col)) direction = "left";

}

else if (direction == "left")

{

if (matrix[row, --col] == 0) matrix[row, col] = index;

if (!IsTraversable(matrix, row, col - 1)) direction = "down";

}

}

}

static bool IsTraversable(int[,] matrix, int row, int col)

{

return row >= 0 && row < matrix.GetLongLength(0) &&

col >= 0 && col < matrix.GetLongLength(1) && matrix[row, col] == 0;

}

// Print the matrix on the console

private static void PrintingMatrix(int[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-2} ", matrix[row, col]); // left aligned

//Console.Write("{0,2} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 2. Maximal sum

Write a program that reads a rectangular matrix of size N x M

and finds in it the square 3 x 3 that has maximal sum of its elements.

\*/

/\* Note: In the file "CheckMaximalSum.txt" there is ready input for easier (faster) testing:

Input: Output:

0, 2, 4, 0, 9, 5 The best platform is:

7, 1, 3, 3, 2, 1 1 3 3

1, 3, 9, 8, 5, 6 3 9 8

4, 6, 7, 9, 1, 0 6 7 9

The maximal sum is: 49

\*/

using System;

class Program

{

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows = ");

int rows = int.Parse(Console.ReadLine());

Console.Write("Number of columns = ");

int cols = int.Parse(Console.ReadLine());

// Create (Allocate) the matrix

int[,] matrix = new int[rows, cols];

// Enter the matrix elements

for (int row = 0; row < rows; row++)

{

for (int col = 0; col < cols; col++)

{

Console.Write("matrix[{0},{1}] = ", row, col);

matrix[row, col] = int.Parse(Console.ReadLine()); ;

}

}

Console.WriteLine("\nThe matrix is as follows:");

PrintingMatrix(matrix);

// Find the maximal sum platform of size 3 x 3

int bestSum = int.MinValue;

int bestRow = 0;

int bestCol = 0;

for (int row = 0; row < matrix.GetLength(0) - 2; row++)

{

for (int col = 0; col < matrix.GetLength(1) - 2; col++)

{

int sum = matrix[row, col] + matrix[row, col + 1] + matrix[row, col + 2] +

matrix[row + 1, col] + matrix[row + 1, col + 1] + matrix[row + 1, col + 2] +

matrix[row + 2, col] + matrix[row + 2, col + 1] + matrix[row + 2, col + 2];

if (sum > bestSum)

{

bestSum = sum;

bestRow = row;

bestCol = col;

}

}

}

// Print the result

Console.WriteLine("\nThe best platform is:");

Console.WriteLine(" {0} {1} {2}",

matrix[bestRow, bestCol], matrix[bestRow, bestCol + 1], matrix[bestRow, bestCol + 2]);

Console.WriteLine(" {0} {1} {2}",

matrix[bestRow + 1, bestCol], matrix[bestRow + 1, bestCol + 1], matrix[bestRow + 1, bestCol + 2]);

Console.WriteLine(" {0} {1} {2}",

matrix[bestRow + 2, bestCol], matrix[bestRow + 2, bestCol + 1], matrix[bestRow + 2, bestCol + 2]);

Console.WriteLine("\nThe maximal sum is: {0}\n", bestSum);

}

// Print the matrix on the console

private static void PrintingMatrix(int[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-2} ", matrix[row, col]); // left aligned

//Console.Write("{0,2} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 3. Sequence n matrix

We are given a matrix of strings of size N x M.

Sequences in the matrix we define as sets of several neighbour elements located on the same line, column or diagonal.

Write a program that finds the longest sequence of equal strings in the matrix.

Example:

matrix result matrix result

ha fifi ho hi s qq s

fo ha hi xx ha, ha, ha pp pp s s, s, s

xxx ho ha xx pp qq s

\*/

/\* Note: In the files "Example1SequenceNMatrix" and "Example2SequenceNMatrix"

there are ready inputs for easier (faster) testing of the given examples.

\*/

using System;

using System.Linq;

class SequenceNMatrix

{

static readonly int[,] directions = { { 0, 1 }, { 1, 1 }, { -1, 1 }, { 1, 0 } };

static void Main()

{

// Read the matrix dimensions

Console.Write("Number of rows = ");

int rows = int.Parse(Console.ReadLine());

Console.Write("Number of columns = ");

int cols = int.Parse(Console.ReadLine());

// Create (Allocate) the matrix

string[,] matrix = new string[rows, cols];

// Enter the matrix elements

for (int row = 0; row < rows; row++)

{

for (int col = 0; col < cols; col++)

{

Console.Write("matrix[{0},{1}] = ", row, col);

matrix[row, col] = Console.ReadLine();

}

}

// Print the matrix on the console

Console.WriteLine("\nThe matrix is as follows:");

PrintingMatrix(matrix);

// Find the longest sequence of equal strings in the matrix

int bestLength = 0;

string bestElement = string.Empty;

FindLongestSequence(matrix, ref bestElement, ref bestLength);

// Print the result

Console.WriteLine("\nThe longest sequence of equal strings in the matrix is:");

Console.WriteLine(string.Join(", ", Enumerable.Repeat(bestElement, bestLength)));

Console.WriteLine();

}

static void FindLongestSequence(string[,] matrix, ref string bestElement, ref int bestLength)

{

for (int row = 0; row < matrix.GetLongLength(0); row++)

{

for (int col = 0; col < matrix.GetLongLength(1); col++)

{

int direction = -1;

while (++direction < 4)

{

int currRow = row + directions[direction, 0];

int currCol = col + directions[direction, 1];

int currentLength = 1;

while (IsTraversable(matrix, row, col, currRow, currCol))

{

currentLength++;

if (currentLength > bestLength)

{

bestLength = currentLength;

bestElement = matrix[row, col];

}

currRow += directions[direction, 0];

currCol += directions[direction, 1];

}

}

}

}

}

static bool IsTraversable(string[,] matrix, int row, int col, int currRow, int currCol)

{

return currRow >= 0 && currRow < matrix.GetLongLength(0) &&

currCol >= 0 && currCol < matrix.GetLongLength(1) &&

matrix[currRow, currCol] == matrix[row, col];

}

// Print the matrix on the console

private static void PrintingMatrix(string[,] matrix)

{

for (int row = 0; row < matrix.GetLength(0); row++) // rows = matrix.GetLength(0)

{

for (int col = 0; col < matrix.GetLength(1); col++) // cols = matrix.GetLength(1)

{

Console.Write("{0,-5} ", matrix[row, col]); // left aligned

//Console.Write("{0,5} ", matrix[row, col]); // right aligned

}

Console.WriteLine();

}

}

}

/\* Problem 4. Binary search

Write a program, that reads from the console an array of N integers and an integer K,

sorts the array and using the method Array.BinSearch() finds the largest number in the array which is ≤ K.

\*/

using System;

class BinarySearch

{

static void Main()

{

Console.Write("Enter an integer number for the length of the array N: ");

int n = int.Parse(Console.ReadLine());

Console.Write("Enter an integer number K: ");

int K = int.Parse(Console.ReadLine());

int[] array = new int[n + 1];

Console.WriteLine("Enter {0} numbers to array:", n);

for (int i = 0; i < n; i++)

{

array[i] = int.Parse(Console.ReadLine());

}

array[n] = K;

// Binary search works only on sorted arrays.

Array.Sort(array);

int indexK = Array.BinarySearch(array, K);

if (indexK == n)

{

Console.WriteLine("\nThe largest number in the array which is <= {0} is: \n{1}", K, array[n - 1]);

}

else if (array[indexK] == array[indexK + 1])

{

Console.WriteLine("\nThe largest number in the array which is <= {0} is: \n{1}", K, array[indexK]);

}

else if (indexK == 0)

{

Console.WriteLine(

"\nThere is no smaller or equal to K = {0} number in the array.\nAll the elements are larger than {0}.", K);

}

else

{

Console.WriteLine(array[indexK - 1]);

}

}

}

/\* Problem 5. Sort by string length

You are given an array of strings.

Write a method that sorts the array by the length of its elements (the number of characters composing them).

\*/

using System;

class SortByStringLength

{

static void Main()

{

Console.Write("Enter N: ");

int n = int.Parse(Console.ReadLine());

string[] array = new string[n];

for (int i = 0; i < n; i++)

{

Console.Write("Enter element {0} : ", i + 1);

array[i] = Console.ReadLine();

}

string exchanger = null;

int indexer = 0;

int counter = 0;

for (int i = 0; i < n; i++)

{

exchanger = array[i];

counter = 0;

for (int j = i; j < n; j++)

{

if (array[j].Length > counter)

{

counter = array[j].Length;

indexer = j;

exchanger = array[j];

}

}

array[indexer] = array[i];

array[i] = exchanger;

}

Array.Reverse(array);

foreach (var item in array)

{

Console.WriteLine(item);

}

}

}

/\* Problem 6.\* Matrix class

Write a class Matrix, to hold a matrix of integers.

Overload the operators for adding, subtracting and multiplying of matrices,

indexer for accessing the matrix content and ToString().

\*/

using System;

using System.Collections.Generic;

public class Matrix

{

private int[,] matrix; //field

public int Rows //property

{

get

{

return this.matrix.GetLength(0);

}

}

public int Columns //property

{

get

{

return this.matrix.GetLength(1);

}

}

public Matrix(int rows, int cols) //constructor

{

this.matrix = new int[rows, cols];

}

public int this[int row, int col] //indexer

{

get

{

return this.matrix[row, col];

}

set

{

this.matrix[row, col] = value;

}

}

public override string ToString() //ToString

{

string answer = null;

for (int i = 0; i < this.Rows; i++)

{

for (int j = 0; j < this.Columns; j++)

{

answer = answer + this.matrix[i, j] + " ";

}

answer = answer + "\n";

}

return answer;

}

public static Matrix Add(Matrix m1, Matrix m2) // I dont Include the check for correct input data

{

Matrix result = new Matrix(m1.Rows, m1.Columns);

for (int i = 0; i < m1.Rows; i++)

{

for (int j = 0; j < m1.Columns; j++)

{

result[i, j] = m1[i, j] + m2[i, j];

}

}

return result;

}

public static Matrix Substract(Matrix m1, Matrix m2) // I dont Include the check for correct input data

{

Matrix result = new Matrix(m1.Rows, m1.Columns);

for (int i = 0; i < m1.Rows; i++)

{

for (int j = 0; j < m1.Columns; j++)

{

result[i, j] = m1[i, j] - m2[i, j];

}

}

return result;

}

public static Matrix Multiply(Matrix m1, Matrix m2) // I dont Include the check for correct input data

{

Matrix result = new Matrix(m1.Rows, m2.Columns);

for (int row = 0; row < m1.Rows; row++)

{

for (int col = 0; col < m2.Columns; col++)

{

int temp = 0;

for (int currentnumbers = 0; currentnumbers < m1.Columns; currentnumbers++)

{

temp = temp + m1[row, currentnumbers] \* m2[currentnumbers, col];

}

result[row, col] = temp;

}

}

return result;

}

public static Matrix operator \*(Matrix m1, Matrix m2) // the static method Multiply remains public, so that both by the operator

//\* and by the method Multiply the activity can be used.

{

return Matrix.Multiply(m1, m2);

}

public static Matrix operator +(Matrix m1, Matrix m2)

{

return Matrix.Add(m1, m2);

}

public static Matrix operator -(Matrix m1, Matrix m2)

{

return Matrix.Substract(m1, m2);

}

}

class ClassMatrixTester

{

static void Main()

{

Matrix testMatrix1 = new Matrix(2, 3);

testMatrix1[0, 0] = 1;

testMatrix1[0, 1] = 2;

testMatrix1[0, 2] = 3;

testMatrix1[1, 0] = 4;

testMatrix1[1, 1] = 5;

testMatrix1[1, 2] = 6;

Matrix testMatrix2 = new Matrix(3, 2);

testMatrix2[0, 0] = 7;

testMatrix2[0, 1] = 8;

testMatrix2[1, 0] = 9;

testMatrix2[1, 1] = 10;

testMatrix2[2, 0] = 11;

testMatrix2[2, 1] = 12;

Matrix testMatrix3 = new Matrix(2, 3);

testMatrix3[0, 0] = 1;

testMatrix3[0, 1] = 2;

testMatrix3[0, 2] = 3;

testMatrix3[1, 0] = 4;

testMatrix3[1, 1] = 5;

testMatrix3[1, 2] = 6;

Matrix resultAdd = Matrix.Add(testMatrix1, testMatrix3);

var add = testMatrix1 + testMatrix3;

Console.WriteLine(add);

Console.WriteLine(resultAdd);

Matrix resultSubstract = Matrix.Substract(testMatrix1, testMatrix3);

var substract = testMatrix1 - testMatrix3;

Console.WriteLine(resultSubstract);

Console.WriteLine(substract);

Matrix resultMultiply = Matrix.Multiply(testMatrix1, testMatrix2);

var multiply = testMatrix1 \* testMatrix2;

Console.WriteLine(resultMultiply);

Console.WriteLine(multiply);

}

}

/\* Problem 7.\* Largest area in matrix

Write a program that finds the largest area of equal neighbour elements in a rectangular matrix and prints its size.

Example:

matrix result

1 3 2 2 2 4

3 3 3 2 4 4

4 3 1 2 3 3

4 3 1 3 3 1

4 3 3 3 1 1 13

Hint: you can use the algorithm Depth-first search or Breadth-first search.

\*/

using System;

using System.Collections.Generic;

class LargestAreaEqualNeighbourElements

{

static int answer = 0;

static int absolutemax = 0;

static int[,] matrix = new int[,] { { 1, 3, 2, 2, 2, 4 }, // stand for check

{ 3, 3, 3, 2, 4, 4 },

{ 4, 3, 1, 2, 3, 3 },

{ 4, 3, 1, 3, 3, 1 },

{ 4, 3, 3, 3, 1, 1 } };

static void Main()

{

for (int i = 0; i < matrix.GetLength(0); i++)

{

for (int j = 0; j < matrix.GetLength(1); j++)

{

FindTheArea(i, j, matrix[i, j]);

answer = 0;

}

}

Console.WriteLine(absolutemax);

}

private static void FindTheArea(int i, int j, int currelement)

{

if ((currelement == 0) || (i < 0) || (i >= matrix.GetLength(0)) || (j < 0) || (j >= matrix.GetLength(1))) //returns if we are out of the matrix or the element is not the same

{

return;

}

if (matrix[i, j] == currelement)

{

matrix[i, j] = 0;

answer++;

if (absolutemax < answer)

{

absolutemax = answer;

}

FindTheArea(i + 1, j, currelement);

FindTheArea(i - 1, j, currelement);

FindTheArea(i, j + 1, currelement);

FindTheArea(i, j - 1, currelement);

matrix[i, j] = currelement;

}

}

}

# Homework: Methods

/\* Problem 1. Say Hello

Write a method that asks the user for his name and prints “Hello, <name>”

Write a program to test this method.

Example:

input output

Peter Hello, Peter!

\*/

using System;

class SayHello

{

static void Main()

{

Console.Write("Enter Your name: ");

string inputName = Console.ReadLine();

PrintName(inputName);

}

private static void PrintName(string name)

{

Console.WriteLine("Hello, {0}!", name);

}

}

/\* Problem 2. Get largest number

Write a method GetMax() with two parameters that returns the larger of two integers.

Write a program that reads 3 integers from the console and prints the largest of them using the method GetMax().

\*/

using System;

class GetLargestNumber

{

static void Main()

{

Console.Write("Enter first integer : ");

int firstNum = int.Parse(Console.ReadLine());

Console.Write("Enter second integer : ");

int secondNum = int.Parse(Console.ReadLine());

Console.Write("Enter third integer : ");

int thirdNum = int.Parse(Console.ReadLine());

int maximum = GetMax(GetMax(firstNum, secondNum), thirdNum);

Console.WriteLine("The largest of the three numbers is: " + maximum);

}

private static int GetMax(int firstNum, int secondNum)

{

return Math.Max(firstNum, secondNum);

//// other way

//return (firstNum > secondNum) ? firstNum : secondNum;

}

//// other way - method with 3 parameters

//static void Main()

//{

// Console.Write("Enter first integer : ");

// int firstNum = int.Parse(Console.ReadLine());

// Console.Write("Enter second integer : ");

// int secondNum = int.Parse(Console.ReadLine());

// Console.Write("Enter third integer : ");

// int thirdNum = int.Parse(Console.ReadLine());

// int maximum = GetMax(firstNum, secondNum, thirdNum);

// Console.WriteLine("The largest of the three numbers is: " + maximum);

//}

//static int GetMax(int one, int two, int three)

//{

// return Math.Max(Math.Max(one, two), three);

//}

}

/\* Problem 3. English digit

Write a method that returns the last digit of given integer as an English word.

Examples:

input output

512 two

1024 four

12309 nine

\*/

using System;

class EnglishDigit

{

static void Main()

{

Console.Write("Enter an integer number: ");

int inputNum = int.Parse(Console.ReadLine());

Console.WriteLine(GetDigit(inputNum));

}

private static string GetDigit(int inputNum)

{

string digitAsWord = null;

string inputAsString = inputNum.ToString();

switch (inputAsString[inputAsString.Length - 1])

{

case '0': digitAsWord = "Zero"; break;

case '1': digitAsWord = "One"; break;

case '2': digitAsWord = "Two"; break;

case '3': digitAsWord = "Three"; break;

case '4': digitAsWord = "Four"; break;

case '5': digitAsWord = "Five"; break;

case '6': digitAsWord = "Six"; break;

case '7': digitAsWord = "Seven"; break;

case '8': digitAsWord = "Eight"; break;

case '9': digitAsWord = "Nine"; break;

}

return digitAsWord;

}

//// other way

//static void Main()

//{

// Console.Write("Enter an integer number: ");

// int inputNum = int.Parse(Console.ReadLine());

// Console.WriteLine((Digit)(inputNum % 10));

//}

//public enum Digit

//{

// Zero = 0,

// One = 1,

// Two = 2,

// Three = 3,

// Four = 4,

// Five = 5,

// Six = 6,

// Seven = 7,

// Eight = 8,

// Nine = 9

//}

}

/\* Problem 4. Appearance count

Write a method that counts how many times given number appears in given array.

Write a test program to check if the method is workings correctly.

\*/

using System;

class HowManyTimesNumberInArray

{

static void Main()

{

Console.Write("Enter an integer number: ");

int numberToCount = int.Parse(Console.ReadLine());

int[] arrayInput = ArrayReader();

int counter = Counter(numberToCount, arrayInput);

Console.WriteLine("Times of appearing: {0}", counter);

}

static int[] ArrayReader()

{

Console.Write("Enter array length: ");

int arrayLength = int.Parse(Console.ReadLine());

int[] arrayInput = new int[arrayLength];

for (int i = 0; i < arrayLength; i++)

{

Console.Write("element[{0}] = ", i);

arrayInput[i] = int.Parse(Console.ReadLine());

}

return arrayInput;

}

static int Counter(int number, params int[] array)

{

int counter = 0;

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

{

if (array[i] == number)

{

counter++;

}

}

return counter;

}

}

/\* Problem 5. Larger than neighbours

Write a method that checks if the element at given position in given array of integers

is larger than its two neighbours (when such exist).

\*/

using System;

class ElementIsBiggerThanNeighbours

{

static void Main()

{

Console.WriteLine("Enter array length: ");

int length = int.Parse(Console.ReadLine());

int[] array = new int[length];

ArrayInput(length, array);

Console.Write("Enter the position of the element - from [1 to {0}]: ", length); // the positions start from 1

int elemPosition = int.Parse(Console.ReadLine());

ElementCompararer(length, array, elemPosition);

}

static void ArrayInput(int length, int[] array)

{

for (int i = 0; i < length; i++)

{

Console.Write("element[{0}] = ", i);

array[i] = int.Parse(Console.ReadLine());

}

}

static void ElementCompararer(int length, int[] array, int elemPosition)

{

if ((elemPosition == 1) || (elemPosition == length))

{

Console.WriteLine("The element on position {0} doesn't have two neighbours.", elemPosition);

}

else

{

if ((array[elemPosition - 1] > array[elemPosition]) && (array[elemPosition - 1] > array[elemPosition - 2]))

{

Console.WriteLine("The element on position {0} is {1}\n and it is bigger than it's two neighbours ({2} and {3}).", elemPosition, array[elemPosition - 1], array[elemPosition - 2], array[elemPosition]);

}

else

{

Console.WriteLine("The element on position {0} is {1}\n and it is NOT bigger than it's two neighbours ({2} and {3}).", elemPosition, array[elemPosition - 1], array[elemPosition - 2], array[elemPosition]);

}

}

}

}

/\* Problem 6. First larger than neighbours

Write a method that returns the index of the first element in array

that is larger than its neighbours, or -1, if there’s no such element.

Use the method from the previous exercise.

\*/

using System;

class IndexOfTheFirstElementBiggerThanItsNeighbours

{

static void Main()

{

int[] array = ArrayInput();

for (int i = 1; i < array.Length - 1; i++) // the first and the last elements do not satisfy the condition - no need to check

{

if (ElementCompararer(array, i) != -1)

{

return;

}

}

Console.WriteLine("-1 (there’s no such element)");

}

public static int[] ArrayInput()

{

Console.Write("Enter array length: ");

int length = int.Parse(Console.ReadLine());

int[] array = new int[length];

for (int i = 0; i < length; i++)

{

Console.Write("element[{0}] = ", i);

array[i] = int.Parse(Console.ReadLine());

}

return array;

}

static int ElementCompararer(int[] array, int elemPosition)

{

if ((array[elemPosition] > array[elemPosition - 1]) && (array[elemPosition] > array[elemPosition + 1]))

{

Console.WriteLine("\nThe index of the first element in the array\n that is larger than its neighbours is: {0}\n", elemPosition);

return elemPosition;

}

else

{

return -1;

}

}

}

/\* Problem 7. Reverse number

Write a method that reverses the digits of given decimal number.

Example:

input output

256 652

123.45 54.321

\*/

using System;

using System.Threading;

using System.Globalization;

class DigitsReverse

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = CultureInfo.InvariantCulture;

Console.Write("Enter a decimal number: ");

double number; // the type is double because of the second example

while (!double.TryParse(Console.ReadLine(), out number))

{

Console.WriteLine("Invalid number.");

Console.Write("Please enter a correct decimal number: ");

}

Console.WriteLine(DigitsReverser(number));

}

static double DigitsReverser(double number)

{

string input = number.ToString();

string output = null;

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

{

output = output + input[input.Length - 1 - i];

}

return double.Parse(output);

}

}

/\* Problem 8. Number as array

Write a method that adds two positive positive numbers represented as arrays of digits

(each array element arr[i] contains a digit; the last digit is kept in arr[0]).

Each of the numbers that will be added could have up to 10 000 digits.

\*/

using System;

using System.Collections.Generic;

using System.Linq;

class NumbersAsArraysSum

{

static void Main()

{

Console.Write("Enter the first positive positive number: ");

string firstNum = Console.ReadLine();

Console.Write("Enter the second positive positive number: ");

string secondNum = Console.ReadLine();

if (IsCorrectNumber(firstNum) && IsCorrectNumber(secondNum))

{

List<int> result = AccumulateTwoNumbers(firstNum, secondNum);

Console.WriteLine("Sum:");

PrintResult(result);

}

else

{

Console.WriteLine("\nYou have entered invalid number(s).\n");

}

}

static bool IsCorrectNumber(string number)

{

return number.All(t => t >= '0' && t <= '9');

}

static List<int> AccumulateTwoNumbers(string firstNum, string secondNum)

{

// Convert string to int[] according to the assignment to represent numbers as array of digits

var firstArray = firstNum.Select(ch => ch - '0').ToArray();

var secondArray = secondNum.Select(ch => ch - '0').ToArray();

Array.Reverse(firstArray);

Array.Reverse(secondArray);

List<int> result = new List<int>(Math.Max(firstArray.Length, secondArray.Length));

int left = 0;

for (int i = 0; i < result.Capacity; i++)

{

int num = (i < firstArray.Length ? firstArray[i] : 0) + (i < secondArray.Length ? secondArray[i] : 0) + left;

result.Add(num % 10);

left = num / 10;

}

if (left > 0) result.Add(left);

return result;

}

static void PrintResult(List<int> result)

{

for (int i = result.Count - 1; i >= 0; i--)

{

Console.Write(result[i]);

}

Console.WriteLine("\n");

}

}

/\* Problem 9. Sorting array

Write a method that return the maximal element in a portion of array of integers starting at given index.

Using it write another method that sorts an array in ascending / descending order.

\*/

using System;

class MaxElementInAPortionOfArray

{

static void Main()

{

Console.Write("Enter array length: ");

int length = int.Parse(Console.ReadLine());

int[] array = new int[length];

ArrayInput(array);

Console.WriteLine("Enter a position (index), from which to start the search [from 0 to {0}]: ", length - 1);

int position = int.Parse(Console.ReadLine());

int maxPosition = 0;

int maxElement = MaxElement(position, array, ref maxPosition, array.Length);//the last var is for the other tasks

Console.WriteLine("The maximum element is {0} at position {1}. ", maxElement, maxPosition);

int[] arrayInputAscending = new int[length];

int[] arrayInputDescending = new int[length];

Array.Copy(array, arrayInputAscending, length);

Array.Copy(array, arrayInputDescending, length);

AscendingSorter(arrayInputAscending);

DescendingSorter(arrayInputDescending);

Console.WriteLine("The array is:");

PrintArray(array);

Console.WriteLine();

Console.WriteLine("The Sorted array in ascending order is:");

PrintArray(arrayInputAscending);

Console.WriteLine();

Console.WriteLine("The Sorted array in descending order is:");

PrintArray(arrayInputDescending);

Console.WriteLine();

}

static void ArrayInput(int[] array)

{

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

{

Console.Write("element[{0}] = ", i);

array[i] = int.Parse(Console.ReadLine());

}

}

static void PrintArray(int[] array)

{

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

{

Console.Write(array[i] + " ");

}

Console.WriteLine();

}

static int MaxElement(int startPosition, int[] array, ref int maxPos, int finishPosition)

{

int maxEl = array[startPosition];

maxPos = startPosition;

for (int i = startPosition; i < finishPosition; i++)

{

if (maxEl < array[i])

{

maxEl = array[i];

maxPos = i;

}

}

return maxEl;

}

static void AscendingSorter(int[] array)

{

int max = 0;

int maxPosition = 0;

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

{

max = MaxElement(0, array, ref maxPosition, array.Length - i);

array[maxPosition] = array[array.Length - 1 - i];

array[array.Length - 1 - i] = max;

}

}

static void DescendingSorter(int[] array)

{

int max = 0;

int maxPosition = 0;

int[] SortedArray = new int[array.Length];

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

{

max = MaxElement(i, array, ref maxPosition, array.Length);

array[maxPosition] = array[i];

array[i] = max;

}

}

}

/\* Problem 10. N Factorial

Write a program to calculate n! for each n in the range [1..100].

Hint: Implement first a method that multiplies a number represented as array of digits by given integer number.

\*/

using System;

using System.Collections.Generic;

using System.Linq;

class NFactorial

{

static void Main()

{

Console.Write("Enter a number N in the range [1..100]: ");

int number = int.Parse(Console.ReadLine());

List<int> factorial = CalculateFactorial(number);

Console.WriteLine("\n{0}! = {1}\n", number, string.Join("", factorial));

}

static List<int> CalculateFactorial(int n)

{

int[] a = { 1 };

int left = 0;

for (int i = 2; i <= n; i++, left = 0)

{

int[] b = i.ToString().Select(ch => ch - '0').ToArray(); // Select => using System.Linq;

int[] c = new int[a.Length + b.Length];

for (int k = a.Length - 1; k >= 0; k--)

{

for (int j = b.Length - 1; j >= 0; j--)

{

c[a.Length - k + b.Length - j - 2] += a[k] \* b[j];

}

}

for (int j = 0; j < c.Length; j++)

{

var digits = c[j] + left;

c[j] = digits % 10;

left = digits / 10;

}

a = c;

Array.Reverse(c);

}

int start = 0;

while (a[start] == 0) start++;

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

for (int i = start; i < a.Length; i++)

{

result.Add(a[i]);

}

return result;

}

}

/\* Problem 11. Adding polynomials

Write a method that adds two polynomials.

Represent them as arrays of their coefficients.

Example:

x2 + 5 = 1x2 + 0x + 5 => {5, 0, 1}

// In the file "example AddingPolynomials" the example is displayed better.

Problem 12. Subtracting polynomials

Extend the previous program to support also subtraction and multiplication of polynomials.

\*/

using Operations\_With\_Polynomials;

namespace Operations\_With\_Polynomials

{

using System;

using System.Collections.Generic;

class PolynomialOperations

{

static void Main(string[] args)

{

Polynomial a;

Polynomial b;

a = new Polynomial("x");

a.AddMononomial(new Mononomial(4, 2));

a.AddMononomial(new Mononomial(-3, 1));

a.AddMononomial(new Mononomial(2, 0));

b = new Polynomial("x");

b.AddMononomial(new Mononomial(5, 2));

b.AddMononomial(new Mononomial(2, 1));

b.AddMononomial(new Mononomial(-7, 0));

PrintResult(a, b, Operation.Addition);

a = new Polynomial("x");

a.AddMononomial(new Mononomial(8, 2));

a.AddMononomial(new Mononomial(2, 1));

b = new Polynomial("x");

b.AddMononomial(new Mononomial(10, 2));

b.AddMononomial(new Mononomial(2, 1));

b.AddMononomial(new Mononomial(-9, 0));

PrintResult(a, b, Operation.Substraction);

a = new Polynomial("x");

a.AddMononomial(new Mononomial(3, 1));

a.AddMononomial(new Mononomial(2, 0));

b = new Polynomial("x");

b.AddMononomial(new Mononomial(9, 2));

b.AddMononomial(new Mononomial(-6, 1));

b.AddMononomial(new Mononomial(4, 0));

PrintResult(a, b, Operation.Multiplication);

}

private static void PrintResult(Polynomial a, Polynomial b, Operation operation)

{

Console.WriteLine("A = " + a);

Console.WriteLine("B = " + b);

switch (operation)

{

case Operation.Addition:

Console.WriteLine("A + B = " + (a + b) + Environment.NewLine);

break;

case Operation.Substraction:

Console.WriteLine("A - B = " + (a - b) + Environment.NewLine);

break;

case Operation.Multiplication:

Console.WriteLine("A \* B = " + (a \* b) + Environment.NewLine);

break;

default:

break;

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Operations\_With\_Polynomials

{

class Polynomial

{

private long maxPower;

private List<Mononomial> mononomials;

private string literal;

public Polynomial(string literal)

{

this.Literal = literal.ToLower();

this.maxPower = -1;

this.Mononomials = new List<Mononomial>();

}

private string Literal

{

get

{

return this.literal;

}

set

{

this.literal = value;

}

}

private uint MaxPower

{

get

{

return (uint)this.maxPower;

}

set

{

this.maxPower = value;

}

}

private List<Mononomial> Mononomials

{

get

{

return this.mononomials;

}

set

{

this.mononomials = value;

}

}

public bool HasAnyMononomials()

{

return (this.maxPower > -1);

}

public void AddMononomial(Mononomial mononomial)

{

this.Mononomials.Add(mononomial);

this.maxPower = Math.Max(this.maxPower, mononomial.Power);

}

public void AddMononomials(ICollection<Mononomial> mononomials)

{

this.Mononomials.AddRange(mononomials);

this.maxPower = Math.Max(this.maxPower, mononomials.Select(x => x.Power).Max());

}

public static Polynomial operator +(Polynomial a, Polynomial b)

{

if (!a.HasAnyMononomials() || !b.HasAnyMononomials())

{

throw new ArgumentNullException("polynomial", "Both polynomials must have at least one mononomial.");

}

uint maxPower = Math.Max(a.MaxPower, b.MaxPower);

List<Mononomial> mononomials = new List<Mononomial>();

for (uint pow = 0; pow <= maxPower; pow++)

{

decimal tempCoeff = 0.0m;

tempCoeff += a.Mononomials.Where(x => x.Power == pow).Select(x => x.Coefficient).ToArray().Sum();

tempCoeff += b.Mononomials.Where(x => x.Power == pow).Select(x => x.Coefficient).ToArray().Sum();

mononomials.Add(new Mononomial(tempCoeff, pow));

}

mononomials = mononomials.Where(x => x != null).ToList();

string literal = (a.Literal == b.Literal) ? b.Literal : a.Literal + b.Literal;

Polynomial result = new Polynomial(literal);

result.AddMononomials(mononomials);

return result;

}

public static Polynomial operator -(Polynomial a, Polynomial b)

{

Polynomial negativeB = new Polynomial(b.Literal);

foreach (var mononomial in b.Mononomials)

{

negativeB.AddMononomial(new Mononomial(-mononomial.Coefficient, mononomial.Power));

}

Polynomial result = a + negativeB;

return result;

}

public static Polynomial operator \*(Polynomial a, Polynomial b)

{

uint maxPower = a.MaxPower + b.MaxPower;

List<Mononomial> mononomials = new List<Mononomial>();

foreach (var mononomialA in a.Mononomials)

{

foreach (var mononomialB in b.Mononomials)

{

decimal tempCoeff = mononomialA.Coefficient;

tempCoeff \*= mononomialB.Coefficient;

uint tempPow = mononomialA.Power + mononomialB.Power;

mononomials.Add(new Mononomial(tempCoeff, tempPow));

}

}

mononomials = mononomials.Where(x => x != null).ToList();

string literal = (a.Literal == b.Literal) ? b.Literal : a.Literal + b.Literal;

Polynomial result = new Polynomial(literal);

result.AddMononomials(mononomials);

return result;

}

private void ClearZeros()

{

for (int mononomial = 0; mononomial < this.Mononomials.Count - 1; mononomial++)

{

uint pow = this.Mononomials[mononomial].Power;

decimal coeff = this.Mononomials[mononomial].Coefficient;

uint nextPow = this.Mononomials[mononomial + 1].Power;

decimal nextCoeff = this.Mononomials[mononomial + 1].Coefficient;

if (pow == nextPow && coeff == -nextCoeff)

{

this.Mononomials.RemoveRange(mononomial, 2);

mononomial -= 2;

}

}

}

public override string ToString()

{

StringBuilder result = new StringBuilder();

this.Mononomials = this.Mononomials.OrderByDescending(x => x.Power).ToList();

ClearZeros();

foreach (var mononomial in this.Mononomials)

{

if (mononomial.Coefficient == 0.0m)

continue;

result.Append(String.Format("{3} {0:0.##}{1}^{2} ", Math.Abs(mononomial.Coefficient),

this.Literal, mononomial.Power, (mononomial.Coefficient >= 0) ? '+' : '-'));

}

return result.ToString().TrimStart(new char[] { '+', ' ' });

}

}

}

namespace Operations\_With\_Polynomials

{

public enum Operation

{

Addition,

Substraction,

Multiplication

}

}

using System;

namespace Operations\_With\_Polynomials

{

class Mononomial

{

private uint power;

private decimal coefficient;

public Mononomial(decimal coefficient, uint power)

{

this.Coefficient = coefficient;

this.Power = power;

}

public uint Power

{

get

{

return this.power;

}

private set

{

this.power = value;

}

}

public decimal Coefficient

{

get

{

return this.coefficient;

}

private set

{

this.coefficient = value;

}

}

}

}

/\* Problem 13. Solve tasks

Write a program that can solve these tasks:

Reverses the digits of a number

Calculates the average of a sequence of integers

Solves a linear equation a \* x + b = 0

Create appropriate methods.

Provide a simple text-based menu for the user to choose which task to solve.

Validate the input data:

The decimal number should be non-negative

The sequence should not be empty

a should not be equal to 0

\*/

using System;

using System.Text;

class MultipleTaskSolver

{

public static void Main()

{

int choice = ChooseTaskToSolve();

if (choice == 1)

{

ReversingDigitsOfNumber();

}

else if (choice == 2)

{

CalculatingAverageOfSequence();

}

else

{

SolvingLinearEquation();

}

}

private static int ChooseTaskToSolve()

{

Console.WriteLine("Choose 1, 2 or 3 for a task to solve: ");

Console.WriteLine("1: for reversing the digits of a number");

Console.WriteLine("2: for calculating the average of sequence of integers");

Console.WriteLine("3: for solving a linear expression of the type ax + b = 0");

int choice = 0;

do

{

int.TryParse(Console.ReadLine(), out choice);

}

while (choice != 1 && choice != 2 && choice != 3);

return choice;

}

public static void ReversingDigitsOfNumber()

{

Console.Write("Enter the number : ");

string number = Console.ReadLine().Trim();

while (number[0] == '-')

{

Console.Write("Enter the number again! It can not be negative : ");

number = Console.ReadLine().Trim();

}

StringBuilder answer = new StringBuilder();

for (int i = number.Length - 1; i >= 0; i--)

{

answer.Append(number[i]);

}

Console.WriteLine(answer.ToString());

}

public static void CalculatingAverageOfSequence()

{

Console.Write("Enter the number of integers: ");

double n = int.Parse(Console.ReadLine());

while (n == 0)

{

Console.WriteLine("The number of integers should not be 0!\n Enter the number of integers again:");

n = int.Parse(Console.ReadLine());

}

double sum = 0;

for (int i = 0; i < n; i++)

{

Console.Write("Enter the {0} integer : ", i);

sum += double.Parse(Console.ReadLine());

}

Console.WriteLine(sum / n);

}

public static void SolvingLinearEquation()

{

Console.Write("Enter coeficient before x : ");

double x = double.Parse(Console.ReadLine());

while (x == 0)

{

Console.WriteLine("the coeficient before x can not be 0! Enter it again: ");

x = double.Parse(Console.ReadLine());

}

Console.Write("Enter the free coeficient : ");

double free = double.Parse(Console.ReadLine());

double answer = (free \* (-1)) / x;

Console.WriteLine(answer);

}

}

/\* Problem 14. Integer calculations

Write methods to calculate minimum, maximum, average, sum and product of given set of integer numbers.

Use variable number of arguments.

/\* Problem 15.\* Number calculations \*

Modify your last program and try to make it work for any number type, not just integer (e.g. decimal, float, byte, etc.)

Use generic method (read in Internet about generic methods in C#).

\*/

using System;

using System.Linq;

namespace Number\_Calculations

{

class NumberCalculations

{

static void Main()

{

int[] intArr = new int[] { -23, -1, 12, 45, 13, 2, 1, 90, 4 };

double[] doubleArr = new double[] { 1.2, 2.3, 3.4, 4.5, 5.6, 6.7, 7.8 };

byte[] byteArr = new byte[] { 2, 1, 5, 3, 2 };

NumberSequence<int> intSequence = new NumberSequence<int>(intArr);

Console.WriteLine(intSequence);

NumberSequence<double> doubleSequence = new NumberSequence<double>(doubleArr);

Console.WriteLine(doubleSequence);

NumberSequence<byte> byteSequence = new NumberSequence<byte>(byteArr);

Console.WriteLine(byteSequence);

}

}

}

using System;

using System.Linq;

namespace Number\_Calculations

{

class NumberSequence<T>

where T : IComparable

{

private T[] numbers;

private T min;

private T max;

private T sum;

private T product;

private T average;

public NumberSequence(T[] numbers)

{

this.numbers = numbers;

GetResults();

}

public T Min

{

get

{

return this.min;

}

private set

{

this.min = value;

}

}

public T Max

{

get

{

return this.max;

}

private set

{

this.max = value;

}

}

public T Sum

{

get

{

return this.sum;

}

private set

{

this.sum = value;

}

}

public T Product

{

get

{

return this.product;

}

private set

{

this.product = value;

}

}

public T Average

{

get

{

return this.average;

}

private set

{

this.average = value;

}

}

private void GetResults()

{

this.Min = this.numbers.Min();

this.Max = this.numbers.Max();

dynamic product = 1;

dynamic sum = 0;

foreach (var number in numbers)

{

product \*= number;

sum += number;

}

this.Product = (T)product;

this.Sum = (T)sum;

this.Average = (T)(this.sum / (dynamic)this.numbers.Length);

}

public override string ToString()

{

return String.Format(@"

[{0}]

MIN: {1, -10} SUM: {3, -15}

MAX: {2, -10} PROD: {4, -15}

AVG: {5}

", String.Join(", ", this.numbers), this.Min, this.Max, this.Sum, this.Product, this.Average);

}

}

}

# Homework: Numeral Systems

/\* Problem 1. Decimal to binary

Write a program to convert decimal numbers to their binary representation.

\*/

/\* Notes: Examples for faster testing:

decimal binary

0 0

3 11

43691 1010101010101011

236476736 1110000110000101100101000000

\*/

using System;

class DecimalToBinary

{

static void Main()

{

long decimalNumber;

Console.Write("Enter an integer number: ");

while (!long.TryParse(Console.ReadLine(), out decimalNumber)) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid input!");

Console.Write("Enter an integer number: ");

}

if (decimalNumber == 0)

{

Console.WriteLine(0);

}

else

{

string binaryNumber = ""; // other way: string binaryNumber = null;

while (decimalNumber > 0)

{

int remainder = (int)decimalNumber % 2;

binaryNumber = remainder + binaryNumber;

decimalNumber /= 2;

}

Console.WriteLine(binaryNumber);

}

}

}

/\* Problem 2. Binary to decimal

Write a program to convert binary numbers to their decimal representation.

\*/

/\* Notes: Examples for faster testing:

binary decimal

0 0

11 3

1010101010101011 43691

1110000110000101100101000000 236476736

\*/

using System;

class BinaryToDecimal

{

static void Main()

{

Console.Write("Enter a binary number (consisted of 0 and 1): ");

string binaryNumber = Console.ReadLine();

long decimalNumber = 0;

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

{

if (binaryNumber[binaryNumber.Length - i - 1] == '0')

{

continue; // bypasses the iteration of the inner-most loop and jumps to the update expression in for loop

}

decimalNumber += (long)Math.Pow(2, i);

}

Console.WriteLine(decimalNumber);

}

}

/\* Problem 3. Decimal to hexadecimal

Write a program to convert decimal numbers to their hexadecimal representation.

\*/

/\* Notes: Examples for faster testing:

decimal hexadecimal

254 FE

6883 1AE3

338583669684 4ED528CBB4

\*/

using System;

class DecimalToHexadecimal

{

static void Main()

{

long decimalNumber;

Console.Write("Enter an integer number: ");

while (!long.TryParse(Console.ReadLine(), out decimalNumber)) // parsing and input check (validating the user data)

{

Console.WriteLine("Invalid input!");

Console.Write("Enter an integer number: ");

}

if (decimalNumber == 0)

{

Console.WriteLine(0);

}

else

{

string hexaNumber = ""; // other way: string hexaNumber = null;

while (decimalNumber > 0)

{

long checkRemainder = (long)decimalNumber % 16; // must be long; when this type is int the result is wrong

string remainder = "";

switch (checkRemainder)

{

case 10: remainder = "A"; break;

case 11: remainder = "B"; break;

case 12: remainder = "C"; break;

case 13: remainder = "D"; break;

case 14: remainder = "E"; break;

case 15: remainder = "F"; break;

default: remainder = checkRemainder.ToString(); break;

}

hexaNumber = remainder + hexaNumber;

decimalNumber /= 16;

}

Console.WriteLine(hexaNumber);

}

}

}

/\* Problem 4. Hexadecimal to decimal

Write a program to convert hexadecimal numbers to their decimal representation.

\*/

/\* Notes: Examples for faster testing:

hexadecimal decimal

FE 254

1AE3 6883

4ED528CBB4 338583669684

\*/

using System;

class HexadecimalToDecimal

{

static void Main()

{

Console.Write("Enter a hexadecimal number: ");

string hexaNumber = Console.ReadLine();

long decimalNumber = 0;

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

{

int multiplier;

switch (hexaNumber[hexaNumber.Length - i - 1])

{

case 'A': multiplier = 10; break;

case 'B': multiplier = 11; break;

case 'C': multiplier = 12; break;

case 'D': multiplier = 13; break;

case 'E': multiplier = 14; break;

case 'F': multiplier = 15; break;

default: multiplier = int.Parse(hexaNumber[hexaNumber.Length - i - 1].ToString()); break;

//default: multiplier = (int)hexaNumber[hexaNumber.Length - i - 1] - 48; break; // other way for default

}

decimalNumber += multiplier \* (long)Math.Pow(16, i);

}

Console.WriteLine(decimalNumber);

}

}

/\* Problem 5. Hexadecimal to binary

Write a program to convert hexadecimal numbers to binary numbers (directly).

\*/

/\* Notes: Examples for faster testing:

hexadecimal binary

FE 11111110

1AE3 0001101011100011

4ED528CBB4 0100111011010101001010001100101110110100

\*/

using System;

using System.Text;

class HexadecimalToBinary

{

static void Main()

{

Console.Write("Enter a hexadecimal number: ");

string hexaNumber = Console.ReadLine();

StringBuilder binaryNumber = new StringBuilder();

foreach (char hexDigit in hexaNumber)

{

switch (hexDigit)

{

case '0': binaryNumber.Append("0000"); break;

case '1': binaryNumber.Append("0001"); break;

case '2': binaryNumber.Append("0010"); break;

case '3': binaryNumber.Append("0011"); break;

case '4': binaryNumber.Append("0100"); break;

case '5': binaryNumber.Append("0101"); break;

case '6': binaryNumber.Append("0110"); break;

case '7': binaryNumber.Append("0111"); break;

case '8': binaryNumber.Append("1000"); break;

case '9': binaryNumber.Append("1001"); break;

case 'A': binaryNumber.Append("1010"); break;

case 'B': binaryNumber.Append("1011"); break;

case 'C': binaryNumber.Append("1100"); break;

case 'D': binaryNumber.Append("1101"); break;

case 'E': binaryNumber.Append("1110"); break;

case 'F': binaryNumber.Append("1111"); break;

default: Console.WriteLine("Invalid hexadecimal number!"); break;

}

}

Console.WriteLine(binaryNumber);

}

}

/\* Problem 6. Binary to hexadecimal

Write a program to convert binary numbers to hexadecimal numbers (directly).

\*/

/\* Notes: Examples for faster testing:

binary hexadecimal

11111110 FE

0001101011100011 1AE3

0100111011010101001010001100101110110100 4ED528CBB4

\*/

using System;

using System.Text;

class BinaryToHexadecimal

{

static void Main()

{

Console.Write("Enter a binary number (consisted of 0 and 1): ");

string binaryNumber = Console.ReadLine();

StringBuilder binary = new StringBuilder(binaryNumber);

bool isNegative = false;

if (binary[0] == '-')

{

isNegative = true;

binary.Remove(0, 1);

}

for (int i = 0, length = binary.Length; i < (4 - length % 4) % 4; i++) //padding leading zeros

{

binary.Insert(0, '0');

}

StringBuilder hexaNumber = new StringBuilder();

StringBuilder word = new StringBuilder("0000");

for (int i = 0; i < binary.Length; i += 4)

{

for (int j = i; j < i + 4; j++)

{

word[j % 4] = binary[j];

}

switch (word.ToString())

{

case "0000": hexaNumber.Append('0'); break;

case "0001": hexaNumber.Append('1'); break;

case "0010": hexaNumber.Append('2'); break;

case "0011": hexaNumber.Append('3'); break;

case "0100": hexaNumber.Append('4'); break;

case "0101": hexaNumber.Append('5'); break;

case "0110": hexaNumber.Append('6'); break;

case "0111": hexaNumber.Append('7'); break;

case "1000": hexaNumber.Append('8'); break;

case "1001": hexaNumber.Append('9'); break;

case "1010": hexaNumber.Append('A'); break;

case "1011": hexaNumber.Append('B'); break;

case "1100": hexaNumber.Append('C'); break;

case "1101": hexaNumber.Append('D'); break;

case "1110": hexaNumber.Append('E'); break;

case "1111": hexaNumber.Append('F'); break;

default: Console.WriteLine("Invalid number"); break;

}

}

if (isNegative)

{

hexaNumber.Insert(0, '-');

}

Console.WriteLine(hexaNumber.ToString());

}

}

/\* Problem 7. One system to any other

Write a program to convert from any numeral system of given base s to any other numeral system of base d (2 ≤ s, d ≤ 16).

\*/

using System;

using System.Linq;

class OneSystemToAnyOther

{

static void Main()

{

int baseX;

Console.Write("Enter the base (of the numeral system)\n of the number You want to convert in the range [2.. 16]: ");

// parsing and input check (validating the user data)

while (!int.TryParse(Console.ReadLine(), out baseX) || baseX < 2 || baseX > 16)

{

Console.WriteLine("Invalid base!");

Console.Write("Enter a base in the range [2.. 16]: ");

}

Console.Write("\nEnter a number [base {0}]: ", baseX);

string number = StringParse();

int baseY;

Console.Write("\nEnter to which base You want to convert the number {0}: ", number);

while (!int.TryParse(Console.ReadLine(), out baseY) || baseX < 2 || baseX > 16)

{

Console.WriteLine("Invalid base!");

Console.Write("Enter a base in the range [2.. 16]: ");

}

string result = ConvertFromDecimalToBaseY(ConvertToDecimal(number.ToArray(), baseX), baseY);

if (IsValidInput(number, result, baseX, baseY))

{

Console.Write("\n{0} [base {1}] converted to [base {2}] => {3}\n\n", number, baseX, baseY, result);

}

else

{

Console.WriteLine("\nYou have entered an invalid number!\n");

}

}

#region Conversion

// Convert number [base X] to number [base 10]

static int ConvertToDecimal(char[] number, int baseX)

{

int result = 0;

for (int i = number.Length - 1, pow = 1; i >= 0; i--, pow \*= baseX)

result += (number[i] >= 'A') ? (number[i] - 'A' + 10) \* pow : (number[i] - '0') \* pow;

return result;

}

// Convert number [base 10] to number [base Y]

static string ConvertFromDecimalToBaseY(int number, int baseY)

{

string result = string.Empty;

while (number > 0)

{

int remainder = number % baseY;

result = remainder >= 10 ? (char)('A' + remainder - 10) + result : remainder + result;

number /= baseY;

}

return result;

}

#endregion

#region Check

static string StringParse()

{

string number = Console.ReadLine();

// Check for incorrect number

if (number.Any(t => t < 'A' && t > 'Z' && t < 'a' && t > 'z' && t < '0' && t > '9'))

{

throw new ArgumentException();

}

number = MakeAllLettersLarge(number);

return number;

}

// To Upper => valid input number

static string MakeAllLettersLarge(string number)

{

char[] digits = number.ToArray();

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

{

digits[i] = char.ToUpper(number[i]);

}

return string.Join("", digits);

}

// Convert the result from BaseY to baseX and compare the new result with the old result (baseX to baseY)

static bool IsValidInput(string number, string result, int baseX, int baseY)

{

return String.CompareOrdinal(ConvertFromDecimalToBaseY(ConvertToDecimal(result.ToArray(), baseY), baseX), number) == 0;

}

#endregion

}

/\* Problem 8. Binary short

Write a program that shows the binary representation of given 16-bit signed integer number (the C# type short).

\*/

/\* Note: Examples for faster testing:

16-bit signed integer binary

-1 1111111111111111

500 0000000111110100

-56 1111111111001000

\*/

using System;

class ShortToBinary

{

static void Main()

{

Console.Write("Enter an integer number in interval [{0}; {1}]: ", short.MinValue, short.MaxValue);

short shortNumber = short.Parse(Console.ReadLine());

Console.WriteLine(ShortToBinaryConvertion(shortNumber));

}

static string ShortToBinaryConvertion(int number)

{

string binaryNumber = string.Empty;

for (int i = 15; i >= 0; i--)

{

binaryNumber = ((number % 2) & 1) + binaryNumber;

number >>= 1;

//if (i % 4 == 0) binary = " " + binary; // for better print

}

return binaryNumber;

}

}

/\* Problem 9.\* Binary floating-point \*

Write a program that shows the internal binary representation

of given 32-bit signed floating-point number in IEEE 754 format (the C# type float).

Example:

number sign exponent mantissa

-27.25 1 10000011 10110100000000000000000

\*/

using System;

using System.Threading;

using System.Globalization;

class IEEE754Format

{

static void Main()

{

Thread.CurrentThread.CurrentCulture = CultureInfo.InvariantCulture;

string input = Console.ReadLine();

string sighn = "0"; //if it is positive it will remain 0

int powOfTwo = 0;

string mantisa = string.Empty;

string exponent = string.Empty;

double rawMantissa = 0.0d;

if (input[0] == '-')

{

sighn = "1"; // generating sigh bit

}

exponent = GenerateExponent(input, ref rawMantissa, ref powOfTwo);

mantisa = GenerateMantissa(input, rawMantissa);

string answer = sighn + " " + exponent + " " + mantisa;

Console.WriteLine();

Console.WriteLine(answer);

Console.WriteLine("- -------- -----------------------");

Console.WriteLine("s exponent mantissa");

Console.WriteLine();

}

private static string GenerateMantissa(string input, double rawMantissa)

{

string mantisaToReturn = null;

rawMantissa -= 1;

while (rawMantissa != 0)

{

rawMantissa \*= 2;

if (rawMantissa < 1)

{

mantisaToReturn += "0";

}

else

{

mantisaToReturn += "1";

rawMantissa -= 1;

}

if (mantisaToReturn.Length == 23)

{

return mantisaToReturn;

}

}

if (mantisaToReturn == null)

{

mantisaToReturn = new string('0', 23);

}

return mantisaToReturn.PadRight(23, '0');

}

private static string GenerateExponent(string input, ref double rawMantissa, ref int powOfTwo)

{

string exponentToReturn = null;

int sighn = 1;

if (input[0] == '-')

{

sighn = -1;

}

int exponentInDecimal = 127;

double givvenInput = double.Parse(input);

givvenInput \*= sighn;

if (givvenInput < 1)

{

powOfTwo = 1;

givvenInput \*= 2;

while (givvenInput < 1)

{

givvenInput \*= 2;

powOfTwo++;

}

exponentInDecimal -= powOfTwo;

}

else

{

powOfTwo = (int)Math.Log(givvenInput, 2);

exponentInDecimal += powOfTwo;

for (int i = 0; i < powOfTwo; i++)

{

givvenInput /= 2;

}

}

rawMantissa = givvenInput;

exponentToReturn = Convert.ToString(exponentInDecimal, 2).PadLeft(8, '0'); //the padding is not actually needed - just for cleaner code views

return exponentToReturn;

}

}

# Homework: Using Classes and Objects

/\* Problem 1. Leap year

Write a program that reads a year from the console and checks whether it is a leap one.

Use System.DateTime.

\*/

using System;

class LeapYear

{

static void Main()

{

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

int year = int.Parse(Console.ReadLine());

bool isLeap = DateTime.IsLeapYear(year);

Console.Write("The year {0} is ", year);

if (!isLeap)

{

Console.Write("NOT ");

}

Console.WriteLine("a leap year.");

}

}

/\* Problem 2. Random numbers

Write a program that generates and prints to the console 10 random values in the range [100, 200].

\*/

using System;

class RandomNumbers

{

static void Main()

{

Random rndGenerator = new Random();

for (int i = 0; i < 10; i++)

{

Console.WriteLine(rndGenerator.Next(100, 201)); // prints to the console random value in the range [100, 200]

}

}

}

/\* Problem 3. Day of week

Write a program that prints to the console which day of the week is today.

Use System.DateTime.

\*/

using System;

class DayOfWeek

{

static void Main()

{

Console.WriteLine(DateTime.Now.DayOfWeek);

}

}

/\* Problem 4. Triangle surface

Write methods that calculate the surface of a triangle by given:

Side and an altitude to it;

Three sides;

Two sides and an angle between them;

Use System.Math.

\*/

/\* Note: Added file "Example-Triangle-Surface" with example for faster testing.

\*/

using System;

class TriangleSurface

{

static void Main()

{

Console.Write("Enter sideA: ");

double sideA = double.Parse(Console.ReadLine());

Console.Write("Enter sideB: ");

double sideB = double.Parse(Console.ReadLine());

Console.Write("Enter sideC: ");

double sideC = double.Parse(Console.ReadLine());

Console.Write("Enter an altitude to sideA: ");

double altitudeToA = double.Parse(Console.ReadLine());

Console.Write("Enter an angle between sideA and sideB: ");

double angleAB = double.Parse(Console.ReadLine());

double area;

area = SideAltitudeTriangleArea(sideA, altitudeToA);

Console.WriteLine("\nSide and an altitude to it: S={0:F1}", area);

area = TreeSidesTriangleArea(sideA, sideB, sideC);

Console.WriteLine("\nThree sides: S={0:F1}", area);

area = SidesAngleTriangleArea(sideA, sideB, angleAB);

Console.WriteLine("\nTwo sides and an angle between them: S={0:F1}\n", area);

}

public static double SideAltitudeTriangleArea(double side, double altitude)

{

return side \* altitude / 2;

}

public static double TreeSidesTriangleArea(double sideA, double sideB, double sideC)

{

double p = (sideA + sideB + sideC) / 2;

return Math.Sqrt(p \* (p - sideA) \* (p - sideB) \* (p - sideC));

}

public static double SidesAngleTriangleArea(double sideA, double sideB, double angleAB)

{

return (sideA \* sideB \* Math.Sin(Math.PI / 180 \* angleAB)) / 2;

}

}

/\* Problem 5. Workdays

Write a method that calculates the number of workdays between today and given date, passed as parameter.

Consider that workdays are all days from Monday to Friday

except a fixed list of public holidays specified preliminary as array.

\*/

using System;

using System.Collections.Generic;

class Workdays

{

static List<DateTime> publicHolydays;

static void Main()

{

publicHolydays = new List<DateTime>()

{new DateTime(2015, 1, 1),new DateTime(2015, 3, 2),new DateTime(2015, 3, 3),

new DateTime(2015, 4, 10), new DateTime(2015, 4, 11),new DateTime(2015, 4, 12),new DateTime(2015, 4, 13),

new DateTime(2015, 5, 1),new DateTime(2015, 5, 6),new DateTime(2015, 9, 21), new DateTime(2015, 9, 22),

new DateTime(2015, 12, 24),new DateTime(2015, 12, 25),new DateTime(2015, 12, 26),new DateTime(2015, 12, 31),};

DateTime currentDate = DateTime.Today; // gets the current date

//Console.WriteLine(Convert.ToString(currentDate.ToShortDateString())); // prints only the current date

Console.Write("Enter a date in format (dd.mm.yyyy): ");

DateTime date = Convert.ToDateTime(Console.ReadLine());

//Console.WriteLine(Convert.ToString(date.ToShortDateString())); // prints only the input date

Console.WriteLine(CountWorkdays(currentDate, date));

}

private static string CountWorkdays(DateTime startDate, DateTime endDate)

{

int workDays = 0;

int holydays = 0;

int weekends = 0;

while (startDate <= endDate)

{

if (publicHolydays.Contains(startDate))

{

holydays++;

}

else if ((int)startDate.DayOfWeek == 6 || (int)startDate.DayOfWeek == 0) // 6 => Suturday; 0 => Sunday

{

weekends++;

}

else

{

workDays++;

}

startDate = startDate.AddDays(1);

}

return String.Format(@"

Holydays: {0}

Weekends: {1}

Workdays: {2}

", holydays, weekends, workDays);

}

}

/\* Problem 6. Sum integers

You are given a sequence of positive integer values written into a string, separated by spaces.

Write a function that reads these values from given string and calculates their sum.

Example:

input output

"43 68 9 23 318" 461

\*/

using System;

class SumIntegers

{

static void Main()

{

Console.WriteLine("Enter integer numbers given in a single line, separated by a space: ");

string line = Console.ReadLine();

// define array of characters-separators ' ' [can be also { ' ', ',', '.' } - for space, comma and dot]

char[] separators = new char[] { ' ' };

/\* define array of strings and using the built-in functionality of the method Split(…) from the class String,

\* split the contents of a given Input (which is String) by array of characters–separators,

which are passed as an argument of the method.

All substrings among which is space will be removed and stored in the splitNumbers array.

\* For removing the empty elements after splitting, we add as a second parameter (argument) the constant

StringSplitOptions.RemoveEmptyEntries so we instruct the method Split(…) to:

\* Return all substrings from the variable that are split by given list of separators.

\* Consider two or more neighboring separators as one. \*/

string[] splitNumbers = line.Split(separators, StringSplitOptions.RemoveEmptyEntries);

//string[] splitNumbers = Console.ReadLine().Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int sum = 0;

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

{

sum += int.Parse(splitNumbers[i]);

}

Console.WriteLine(sum);

}

}

/\* Problem 7.\* Arithmetical expressions \*

Write a program that calculates the value of given arithmetical expression.

The expression can contain the following elements only:

Real numbers, e.g. 5, 18.33, 3.14159, 12.6

Arithmetic operators: +, -, \*, / (standard priorities)

Mathematical functions: ln(x), sqrt(x), pow(x,y)

Brackets (for changing the default priorities): (, )

Examples:

input output

(3+5.3) \* 2.7 - ln(22) / pow(2.2 -1.7) ~10.6

pow(2, 3.14) \* (3 - (3 \* sqrt(2) - 3.2) + 1.5\*0.3) ~21.22

Hint: Use the classical Shunting-yard algorithm and Reverse Polish notation.

\*/

using System;

using System.Threading;

using System.Globalization;

using System.Collections;

class ArithmeticalExpressions

{

static readonly string[] numbers = { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", ".", "~" };

static readonly string[] operators = { "+", "-", "\*", "/" };

static bool IsNumber(string character)

{

bool charIsNumber = false;

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

{

if (numbers[i] == character)

{

charIsNumber = true;

}

}

return charIsNumber;

}

static bool IsOperator(string character)

{

bool charIsOperator = false;

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

{

if (operators[i] == character)

{

charIsOperator = true;

}

}

return charIsOperator;

}

static int Precedence(string operatorChar)

{

if (operatorChar == "+" || operatorChar == "-")

{

return 1;

}

else

{

return 2;

}

}

static decimal ConvertToDecimal(string input)

{

string output = string.Empty;

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

{

if (input[i] == '~')

{

output += "-";

}

else

{

output += input[i];

}

}

decimal result = decimal.Parse(output);

return result;

}

static string AddPar(string input)

{

string[] separatedString = input.Split(' ');

string result = string.Empty;

bool isFunction = false;

int leftCount = 0;

int rightCount = 0;

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

{

if (separatedString[i] == "ln" || separatedString[i] == "sqrt" || separatedString[i] == "pow")

{

isFunction = true;

result += " ( ";

}

if (isFunction && separatedString[i] == "(")

{

leftCount++;

}

if (isFunction && separatedString[i] == ")")

{

rightCount++;

}

result += separatedString[i] + " ";

if (isFunction && rightCount == leftCount && leftCount != 0)

{

isFunction = false;

leftCount = 0;

rightCount = 0;

result += " ) ";

}

}

return result;

}

static void Main()

{

Thread.CurrentThread.CurrentCulture = CultureInfo.InvariantCulture;

Stack operators = new Stack();

Queue output = new Queue();

Console.Write("Input: ");

string input = Console.ReadLine();

input += " ";

string convertedInput = input[0].ToString();

string previousChar = " ";

string currentChar = " ";

if (IsNumber(input[0].ToString()))

{

previousChar = "number";

}

if (IsOperator(input[0].ToString()))

{

previousChar = "operator";

}

for (int i = 1; i < input.Length - 4; i++)

{

if (input[i].ToString() == ",")

{

currentChar = "comma";

}

if (IsNumber(input[i].ToString()))

{

currentChar = "number";

}

if (IsOperator(input[i].ToString()))

{

currentChar = "operator";

}

if (input[i].ToString() == "(" || input[i].ToString() == ")")

{

currentChar = " ";

convertedInput += " ";

}

if (input[i].ToString() == " ")

{

currentChar = " ";

}

if (input[i].ToString() + input[i + 1].ToString() == "ln")

{

currentChar = "function";

convertedInput += " ln ";

i += 2;

}

if (input[i].ToString() + input[i + 1].ToString() + input[i + 2].ToString() + input[i + 3].ToString() == "sqrt")

{

currentChar = "function";

convertedInput += " sqrt ";

i += 4;

}

if (input[i].ToString() + input[i + 1].ToString() + input[i + 2].ToString() + input[i + 3].ToString() == "sqrt")

{

currentChar = "function";

convertedInput += " sqrt ";

i += 4;

}

if (input[i].ToString() + input[i + 1].ToString() + input[i + 2].ToString() == "pow")

{

currentChar = "function";

convertedInput += " pow ";

i += 3;

}

if (currentChar == previousChar)

{

convertedInput += input[i];

}

else if (input[i].ToString() == " ")

{

currentChar = " ";

convertedInput += " " + input[i];

}

else

{

convertedInput += " " + input[i];

}

previousChar = currentChar;

}

convertedInput = AddPar(convertedInput);

//Console.WriteLine(convertedInput);

//separate the input into tokens

string[] separatedInput = convertedInput.Split(' ');

object current;

//convert to RPN

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

{

if (separatedInput[i] == string.Empty)

{

continue;

}

else if (IsNumber(separatedInput[i].Substring(0, 1))) //add number to output

{

output.Enqueue(separatedInput[i]);

}

else if (separatedInput[i] == "ln" || separatedInput[i] == "sqrt" || separatedInput[i] == "pow") //add function to stack

{

operators.Push(separatedInput[i]);

}

else if (separatedInput[i] == ",")

{

if (operators.Contains("(") && operators.Count != 0)

{

while (operators.Peek() != "(")

{

current = operators.Pop();

output.Enqueue(current);

//i++;

}

}

else

{

Console.WriteLine("Separator ',' was misplaced or parentheses () were mismatched");

return;

}

}

else if (IsOperator(separatedInput[i].Substring(0, 1)))

{

if (operators.Count != 0)

{

string currentOperator = operators.Peek().ToString();

int firstOperatorPrecedence = Precedence(separatedInput[i]);

int secondOperatorPrecedence = Precedence(operators.Peek().ToString());

bool isTrue = (IsOperator(currentOperator)) && (firstOperatorPrecedence <= secondOperatorPrecedence);

while (isTrue && operators.Count != 0)

{

current = operators.Pop();

output.Enqueue(current);

if (operators.Count == 0)

{

break;

}

//i++;

currentOperator = operators.Peek().ToString();

firstOperatorPrecedence = Precedence(separatedInput[i]);

secondOperatorPrecedence = Precedence(operators.Peek().ToString());

isTrue = (IsOperator(currentOperator)) && (firstOperatorPrecedence <= secondOperatorPrecedence);

}

}

operators.Push(separatedInput[i]);

}

else if (separatedInput[i] == "(")

{

operators.Push("(");

}

else if (separatedInput[i] == ")")

{

if (operators.Contains("(") && operators.Count != 0)

{

while (operators.Peek() != "(")

{

current = operators.Pop();

output.Enqueue(current);

//i++;

}

if (operators.Count != 0)

{

current = operators.Pop();

}

if (operators.Count != 0)

{

while (operators.Peek() == "ln" || operators.Peek() == "sqrt" || operators.Peek() == "pow")

{

current = operators.Pop();

output.Enqueue(current);

//i++;

}

}

}

else

{

Console.WriteLine("Parentheses () were mismatched.");

return;

}

}

else

{

Console.WriteLine("Invalid input!");

return;

}

}

while (operators.Count != 0)

{

if (operators.Peek() == "(" || operators.Peek() == ")")

{

Console.WriteLine("Parentheses () were mismatched.");

return;

}

else

{

current = operators.Pop();

output.Enqueue(current);

}

}

//print RPN

string RPV = string.Empty;

while (output.Count != 0)

{

RPV += (output.Dequeue() + " ");

}

//Console.WriteLine(RPV);

string[] separatedRPV = RPV.Split(' ');

Stack calculations = new Stack();

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

{

if (separatedRPV[i] == string.Empty)

{

continue;

}

if (IsNumber(separatedRPV[i].Substring(0, 1)) || separatedRPV[i].Substring(0, 1) == ".")

{

calculations.Push(separatedRPV[i]);

}

if (IsOperator(separatedRPV[i]))

{

decimal firstNumber = 0;

decimal secondNumber = 0;

try

{

firstNumber = ConvertToDecimal(calculations.Pop().ToString());

secondNumber = ConvertToDecimal(calculations.Pop().ToString());

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

decimal result = 0;

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

{

try

{

result = firstNumber + secondNumber;

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

if (separatedRPV[i] == "-")

{

try

{

result = secondNumber - firstNumber;

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

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

{

try

{

result = firstNumber \* secondNumber;

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

if (separatedRPV[i] == "/")

{

try

{

result = secondNumber / firstNumber;

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

calculations.Push(result.ToString());

}

if (separatedRPV[i] == "ln")

{

try

{

decimal result = 0;

decimal number = ConvertToDecimal(calculations.Pop().ToString());

result = (decimal)Math.Log((double)number);

calculations.Push(result);

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

if (separatedRPV[i] == "sqrt")

{

try

{

decimal result = 0;

decimal number = ConvertToDecimal(calculations.Pop().ToString());

result = (decimal)Math.Sqrt((double)number);

calculations.Push(result);

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

if (separatedRPV[i] == "pow")

{

try

{

decimal firstNumber = ConvertToDecimal(calculations.Pop().ToString());

decimal secondNumber = ConvertToDecimal(calculations.Pop().ToString());

decimal result = 0;

result = (decimal)Math.Pow((double)secondNumber, (double)firstNumber);

calculations.Push(result);

}

catch (OverflowException)

{

Console.WriteLine("Result is too big to fit in decimal, try with smaller numbers!");

return;

}

}

}

try

{

Console.Write("\n\rResult: ");

Console.WriteLine(calculations.Pop());

Console.WriteLine();

}

catch (InvalidOperationException)

{

Console.WriteLine("Invalid input!");

Console.WriteLine();

}

}

}

# Homework: Strings and Text Processing

/\* Problem 1. Strings in C#.

Describe the strings in C#.

What is typical for the string data type?

Describe the most important methods of the String class.

\*/

using System;

using System.Text;

class StringsInCSharp

{

static void Main()

{

Console.BufferHeight = 80;

Console.WriteLine("1) Describe the strings in C#:\n");

DescribingStrings();

Console.WriteLine("2) What is typical for the string data type?\n");

TypicalForStringDataType();

Console.WriteLine("3) Describe the most important methods of the String class:\n");

ImportantMethodsOfTheStringClass();

}

private static void DescribingStrings()

{

StringBuilder sb = new StringBuilder();

sb.AppendLine("Strings are immutable (read-only) sequences of characters,\n represented by the string data type in C# (System.String).");

sb.AppendLine("Use Unicode to support multiple languages and alphabets.");

sb.AppendLine("Declared by the keyword string in C#.");

sb.AppendLine("Strings are stored in the dynamic memory (managed heap).");

Console.WriteLine(sb);

}

private static void TypicalForStringDataType()

{

StringBuilder sb = new StringBuilder();

sb.AppendLine("System.String is reference type.");

sb.AppendLine("The value of the String object is the content of the sequential collection\n and that value is immutable (it is read-only).");

sb.AppendLine("The maximum size of a String object in memory is 2 GB, or about 1 billion characters. ");

sb.AppendLine("This type allows us to manipulate character data through different methods and properties.");

sb.AppendLine("String objects are like arrays of characters (char[]).");

sb.AppendLine("Elements can be accessed directly by index.\n The index is in the range [0...Length-1].");

sb.AppendLine("Have fixed length (String.Length).");

Console.WriteLine(sb);

}

private static void ImportantMethodsOfTheStringClass()

{

StringBuilder sb = new StringBuilder();

sb.AppendLine("Most important methods of the String class are:");

sb.AppendLine("1. Length => Get length of the string");

sb.AppendLine("2. str[index] => Indexer in C#; gives access to the char value at given position");

sb.AppendLine("3. string.Compare(str1, str2) or str1.CompareTo(str2) => Comparison of the two strings.\n Returns 0 if str1 equals str2. Returns -1 if str1 is before str2. Returns 1 if str1 is after str2.");

sb.AppendLine("4. str.IndexOf(strSearch) => Returns the start index of the substring strSearch in str\n by its first occurrences");

sb.AppendLine("5. str.LastIndexOf(strSearch) => Returns the start index of the substring strSearch in str\n by its last occurrences");

sb.AppendLine("6. Substring(startIndex, length) => Returns the substring from startIndex with given length");

sb.AppendLine("7. Replace(oldStr, newStr) => Replaces all occurrences of given string oldStr with newStr");

sb.AppendLine("8. Remove(startIndex, length) => Deletes part of a string and produces new string as result;\n removes the characters from startIndex to given length");

sb.AppendLine("9. ToUpper() => Convert to uppercase");

sb.AppendLine("10. ToLower() => Convert to lowercase ");

sb.AppendLine("11. Trim() => Remove leading and trailing whitespace");

sb.AppendLine("12. ToString() => Convert objects to strings ");

sb.AppendLine("13. Split() => Split string by given char/s");

sb.AppendLine("14. string.Concat(str1, str2) => Concatenates 2 strings");

Console.WriteLine(sb);

}

}

/\* Problem 2. Reverse string

Write a program that reads a string, reverses it and prints the result at the console.

Example:

input output

sample elpmas

\*/

using System;

using System.Text;

using System.Linq;

class ReverseString

{

public static string ReversingString(string str)

{

var sb = new StringBuilder(); // using System.Text;

// StringBuilder sb = new StringBuilder(str.Length);

for (int i = str.Length - 1; i >= 0; i--)

{

sb.Append(str[i]);

}

return sb.ToString();

}

static void Main()

{

string str = Console.ReadLine();

string reversed = ReversingString(str);

Console.WriteLine(reversed);

//// other way

//Console.WriteLine(new string(str.ToCharArray().Reverse().ToArray())); // using System.Linq;

}

}

/\* Problem 3. Correct brackets

Write a program to check if in a given expression the brackets are put correctly.

Example of correct expression: ((a+b)/5-d).

Example of incorrect expression: )(a+b)).

\*/

using System;

class CorrectBrackets

{

static void Main()

{

/\* Use counting of the brackets:

\* For an opening bracket increase the counter by 1 and for closing bracket decrease it by 1.

\* Watch the counter not to become a negative number and always ends with 0.

\*/

string expression = Console.ReadLine();

bool correctBrackets = true;

int counter = 0;

for (int index = 0; index < expression.Length; index++)

{

char ch = expression[index];

if (ch == '(')

{

counter++;

}

else if (ch == ')')

{

counter--;

if (counter < 0)

{

correctBrackets = false;

break;

}

}

}

if (counter != 0)

{

correctBrackets = false;

}

Console.WriteLine("Are the brackets correct?\n" + correctBrackets);

}

}

/\* Problem 3. Correct brackets - Solution with Stack data structure

Write a program to check if in a given expression the brackets are put correctly.

Example of correct expression: ((a+b)/5-d).

Example of incorrect expression: )(a+b)).

\*/

using System;

using System.Collections.Generic;

class CorrectBracketsStackSolution

{

static void Main()

{

string expression = Console.ReadLine();

Stack<int> stack = new Stack<int>(); // need using System.Collections.Generic;

/\* The stack is a data structure, which implements the behavior "last in – first out" (LIFO).

The elements could be added and removed only on the top of the stack. \*/

bool correctBrackets = true;

for (int index = 0; index < expression.Length; index++)

{

char ch = expression[index];

if (ch == '(')

{

stack.Push(index); // adds a new element on the top of the stack

}

else if (ch == ')')

{

if (stack.Count == 0) // returns the count of elements in the stack

{

correctBrackets = false;

break;

}

stack.Pop(); // returns the highest element and removes it from the stack

}

}

if (stack.Count != 0)

{

correctBrackets = false;

}

Console.WriteLine("Are the brackets correct?\n" + correctBrackets);

}

}

/\* Problem 4. Sub-string in text

Write a program that finds how many times a sub-string is contained in a given text (perform case insensitive search).

Example:

The target sub-string is "in"

The text is as follows: We are living in an yellow submarine. We don't have anything else. inside the submarine is very tight. So we are drinking all the day. We will move out of it in 5 days.

The result is: 9

\*/

using System;

class SubstringInText

{

static void Main()

{

// for easier testing

string str = "We are living in an yellow submarine. We don't have anything else. inside the submarine is very tight. So we are drinking all the day. We will move out of it in 5 days.";

string targetSubstr = "in";

//// the real input

//string str = Console.ReadLine();

//targetSubstr = Console.ReadLine();

str = str.ToLower(); // to perform insensitive search

int count = 0;

int index = 0;

while (true)

{

int found = str.IndexOf(targetSubstr, index); /\* searching for the targerSubStr in str starting from index

\* return index or -1 if no found \*/

if (found < 0) // (found == -1)

{

break;

}

count++;

index = found + 1;

}

Console.WriteLine(count);

//// other way

//int counter = 0;

//int indexer = str.IndexOf(targetSubstr); /\* searching for the targerSubStr in str

// \* return index or -1 if no found \*/

//while (indexer != -1)

//{

// counter++;

// indexer = str.IndexOf(targetSubstr, indexer + 1);

//}

//Console.WriteLine(counter);

}

}

/\* Problem 5. Parse tags

You are given a text. Write a program that changes the text in all regions surrounded by the tags <upcase> and </upcase> to upper-case.

The tags cannot be nested.

Example:

We are living in a <upcase>yellow submarine</upcase>. We don't have <upcase>anything</upcase> else.

The expected result: We are living in a YELLOW SUBMARINE. We don't have ANYTHING else.

\*/

using System;

using System.Text;

using System.Text.RegularExpressions;

class ParseTags

{

static void Main()

{

// for easier testing

string text = "We are living in a <upcase>yellow submarine</upcase>. We don't have <upcase>anything</upcase> else.";

//// the real input

//string str = Console.ReadLine();

StringBuilder sb = new StringBuilder();

int toUpper = -1; //checks if we have to enter UPPERCASE MODE

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

{

if (text[i] == '<')

{

i++;

toUpper = toUpper \* (-1); // an opening tag => start UPPER

while (text[i] != '>') // disregard all of the tag's content

{

i++;

}

}

else

{

if (toUpper == 1)

{

sb.Append(text[i].ToString().ToUpper());

}

else

{

sb.Append(text[i]);

}

}

}

Console.WriteLine(sb.ToString());

////other way

//Console.WriteLine(Regex.Replace(str, "<upcase>(.\*?)</upcase>", word => word.Groups[1].Value.ToUpper()));

}

}

/\* Problem 6. String length

Write a program that reads from the console a string of maximum 20 characters.

If the length of the string is less than 20, the rest of the characters should be filled with \*.

Print the result string into the console.

\*/

using System;

using System.Text;

class StringLength

{

static void Main()

{

Console.WriteLine("Enter a string of maximum 20 characters:");

string str = Console.ReadLine();

while (str.Length > 20)

{

Console.WriteLine("Incorrect string!");

Console.WriteLine("Enter a string of maximum 20 characters:");

str = Console.ReadLine();

}

// first way

str = str.PadRight(20, '\*');

Console.WriteLine(str); /\* Returns a new string that left-aligns the characters in this string

\* by padding them on the right with a specified Unicode character,

\* for a specified total length. \*/

// second way

//StringBuilder sb = new StringBuilder();

//for (int i = 0; i < str.Length; i++) // copy the str in sb

//{

// sb.Append(str[i]);

//}

////int counter = 0; // other way for copy the str in sb

////while (counter < str.Length)

////{

//// sb.Append(str[counter]);

//// counter++;

////}

//if (sb.Length < 20)

//{

// sb.Append(new string('\*', 20 - sb.Length));

//}

//Console.WriteLine(sb.ToString());

}

}

/\* Problem 7. Encode/decode

Write a program that encodes and decodes a string using given encryption key (cipher).

The key consists of a sequence of characters.

The encoding/decoding is done by performing XOR (exclusive or) operation

over the first letter of the string with the first of the key, the second – with the second, etc.

When the last key character is reached, the next is the first.

\*/

/\* Note: Example for faster testing:

Input: Output:

text: Test 5↕▬

cipher code: "ab" \u0035\u0007\u0012\u0016

Test

\*/

using System;

using System.Text;

class EncodeDecode

{

static void Main()

{

// for faster testing:

string text = "Test";

string key = "ab";

//// the real input

//Console.Write("Enter a string:");

//string text = Console.ReadLine();

//Console.Write("Enter an encryption key (cipher):");

//string key = Console.ReadLine();

StringBuilder encryption = EnDeCryption(text, key);

Console.WriteLine("\nThe result of encryption is:");

Console.WriteLine(encryption); // print the result of encryption

Console.WriteLine("The result of encryption as a series of Unicode escape characters \\xxxx is:");

foreach (var ch in encryption.ToString())

{

Console.Write("\\u{0:x4}", (int)ch); // print the result of encryption as a series of Unicode escape characters \xxxx

}

string encryptText = encryption.ToString();

StringBuilder decryption = EnDeCryption(encryptText, key);

Console.WriteLine("\n\nThe result of decryption is:");

Console.WriteLine("{0}\n", decryption);

}

private static StringBuilder EnDeCryption(string message, string cipher)

{

StringBuilder enDeCrypt = new StringBuilder();

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

{

enDeCrypt.Append((char)(message[i] ^ cipher[i % cipher.Length]));

}

return enDeCrypt;

}

}

/\* Problem 8. Extract sentences

Write a program that extracts from a given text all sentences containing given word.

Example:

The word is: "in"

The text is: We are living in a yellow submarine. We don't have anything else. Inside the submarine is very tight. So we are drinking all the day. We will move out of it in 5 days.

The expected result is: We are living in a yellow submarine. We will move out of it in 5 days.

Consider that the sentences are separated by . and the words – by non-letter symbols.

\*/

using System;

class ExtractSentences

{

static void Main()

{

// input for faster testing

string text = "We are living in a yellow submarine. We don't have anything else. Inside the submarine is very tight. So we are drinking all the day. We will move out of it in 5 days.";

string word = "in";

//// the real input

//Console.WriteLine("Enter a text:");

//string text = Console.ReadLine();

//Console.WriteLine("Enter a word:");

//string word = Console.ReadLine();

string[] sentences = text.Split(new char[]{'.'}, StringSplitOptions.RemoveEmptyEntries);

foreach (string sentence in sentences)

{

string checkingSentence = sentence.ToLower();

int index = checkingSentence.IndexOf(word.ToLower() + " ");

if (index > -1)

{

Console.Write(sentence.Trim() + ". ");

}

}

Console.WriteLine();

}

}

/\* Problem 9. Forbidden words

We are given a string containing a list of forbidden words and a text containing some of these words.

Write a program that replaces the forbidden words with asterisks.

Example text: Microsoft announced its next generation PHP compiler today.

It is based on .NET Framework 4.0 and is implemented as a dynamic language in CLR.

Forbidden words: PHP, CLR, Microsoft

The expected result: \*\*\*\*\*\*\*\*\* announced its next generation \*\*\* compiler today. It is based on .NET Framework 4.0 and is implemented as a dynamic language in \*\*\*.

\*/

using System;

using System.Text;

class ForbiddenWords

{

static void Main()

{

string[] forbiddenWords = new string[] { "PHP", "CLR", "Microsoft" };

string text = "Microsoft announced its next generation PHP compiler today. It is based on .NET Framework 4.0 and is implemented as a dynamic language in CLR.";

StringBuilder asterisksReplacedText = new StringBuilder();

asterisksReplacedText.Append(text);

foreach (string word in forbiddenWords)

{

asterisksReplacedText.Replace(word, new string('\*', word.Length)); // replace word with new string('\*', word.Length)

}

Console.WriteLine(asterisksReplacedText);

}

}

/\* Problem 10. Unicode characters

Write a program that converts a string to a sequence of C# Unicode character literals.

Use format strings.

Example:

input output

Hi! \u0048\u0069\u0021

\*/

using System;

using System.Text;

class UnicodeCharacters

{

static void Main()

{

string input = Console.ReadLine();

StringBuilder answer = new StringBuilder();

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

{

answer.AppendFormat("\\u{0:X4}", (int)input[i]);

}

Console.WriteLine(answer.ToString());

}

}

/\* Problem 11. Format number

Write a program that reads a number and prints

it as a decimal number, hexadecimal number, percentage and in scientific notation.

Format the output aligned right in 15 symbols.

\*/

using System;

class FormatNumber

{

static void Main()

{

int number = int.Parse(Console.ReadLine());

Console.WriteLine("{0,15:D}", number); // Decimal

Console.WriteLine("{0,15:X}", number); // Hexadecimal

Console.WriteLine("{0,15:P}", number); // Percentage

Console.WriteLine("{0,15:E}", number); // Scientific notation

}

}

/\* Problem 12. Parse URL

Write a program that parses an URL address given in the format:

[protocol]://[server]/[resource] and extracts from it the [protocol], [server] and [resource] elements.

Example:

URL Information

http://telerikacademy.com/Courses/Courses/Details/212 [protocol] = http

[server] = telerikacademy.com

[resource] = /Courses/Courses/Details/212

\*/

using System;

class ParseURL

{

static void Main()

{

// input for faster testing

string uRL = "http://telerikacademy.com/Courses/Courses/Details/212";

//// the real input

//string uRL = Console.ReadLine();

/\* Searching for the respective splitters – two slashes for a protocol and

\* one slash as a separator between the server and the resource \*/

int indexOfProtocol = uRL.IndexOf("://"); //the start index of the substring "://" in uRL by its first occurrences

string protocol = uRL.Substring(0, indexOfProtocol); // the substring from 0 with length = indexOfProtocol

protocol = "[protocol] = " + protocol;

//the start index of the substring "/" starting from (indexOfProtocol + 3) in uRL by its first occurrences

int indexOfServer = uRL.IndexOf("/", indexOfProtocol + 3);

string server = uRL.Substring(indexOfProtocol + 3, indexOfServer - indexOfProtocol - 3);

server = "[server] = " + server;

string resource = uRL.Substring(indexOfServer);

resource = "[resource] = " + resource;

Console.WriteLine(protocol);

Console.WriteLine(server);

Console.WriteLine(resource);

}

}

/\* Problem 13. Reverse sentence

Write a program that reverses the words in given sentence.

Example:

input output

C# is not C++, not PHP and not Delphi! Delphi not and PHP, not C++ not is C#!

\*/

using System;

using System.Text;

class ReverseSentence

{

static void Main()

{

string sentence = "C# is not C++, not PHP and not Delphi!"; // input for faster testing

//string sentence = Console.ReadLine(); // the real input

var revercedSentence = new StringBuilder(sentence.Length);

char lastSign = sentence[sentence.Length - 1];

string[] allWords = sentence.Substring(0, sentence.Length - 1).Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

Array.Reverse(allWords);

Console.WriteLine("Reversed: {0}{1}\n", string.Join(" ", allWords), lastSign); // Delphi not and PHP not C++, not is C#!

}

}

/\* Problem 14. Word dictionary

A dictionary is stored as a sequence of text lines containing words and their explanations.

Write a program that enters a word and translates it by using the dictionary.

Sample dictionary:

input output

.NET platform for applications from Microsoft

CLR managed execution environment for .NET

namespace hierarchical organization of classes

\*/

using System;

using System.Collections.Generic;

class WordDictionary

{

static void Main()

{

Dictionary<string, string> dictionary = new Dictionary<string, string>() //using System.Collections.Generic;

{

{".NET", "platform for applications from Microsoft"},

{"CLR", "managed execution environment for .NET"},

{"NAMESPACE", "hierarchical organization of classes"}

};

//// void Add(K, V) => adds a new pair (key and a value) to the dictionarty => throws an exception if the key exists

//dictionary.Add(".NET", "platform for applications from Microsoft");

//dictionary.Add("CLR", "managed execution environment for .NET");

//dictionary.Add("namespace", "hierarchical organization of classes");

Console.WriteLine("Dictionary words: {0}\n", string.Join(", ", dictionary.Keys));

Console.WriteLine("Enter a word to see its translation or type \"exit\" to leave: ");

// By default the strings in .NET are case sensitive (the compiler distinguishes uppercase from lowercase letters)

string input = Console.ReadLine().ToUpper();

while (input != "EXIT")

{

// bool ContainsKey(K) => check if there is an ordered pair with this key in the dictionary

if (dictionary.ContainsKey(input))

{

Console.WriteLine(dictionary[input]);

}

else

{

Console.WriteLine("There is not such word in the dictionary.");

}

Console.WriteLine("Enter a word to see its translation or type EXIT to leave: ");

input = Console.ReadLine().ToUpper();

}

}

}

/\* Problem 15. Replace tags

Write a program that replaces in a HTML document given as string all the tags

<a href="…">…</a> with corresponding tags [URL=…]…[/URL].

Example:

input:

<p>Please visit <a href="http://academy.telerik.com">our site</a> to choose a training course. Also visit <a href="www.devbg.org">our forum</a> to discuss the courses.</p>

output:

<p>Please visit [URL=http://academy.telerik.com]our site[/URL] to choose a training course. Also visit [URL=www.devbg.org]our forum[/URL] to discuss the courses.</p>

\*/

using System;

using System.Text;

using System.Text.RegularExpressions;

class ReplaceTags

{

static void Main()

{

string hTMLDoc = "<p>Please visit <a href=\"http://academy.telerik.com\">our site</a> to choose a training course. Also visit <a href=\"www.devbg.org\">our forum</a> to discuss the courses.</p>";

StringBuilder tagReplaced = new StringBuilder();

string[] messages = hTMLDoc.Split(new string[] { "<a href", "</a>" }, StringSplitOptions.None); // The return value includes array elements that contain an empty string

foreach (var item in messages)

{

int indexOfLink = item.IndexOf("=\"");

if (indexOfLink >= 0)

{

int endIndex = item.IndexOf("\">");

tagReplaced.Append("[URL=");

tagReplaced.Append(item.Substring(indexOfLink + 2, endIndex - indexOfLink - 2));

tagReplaced.Append("]");

tagReplaced.Append(item.Substring(endIndex + 2, item.Length - endIndex - 2));

tagReplaced.Append("[/URL]");

}

else

{

tagReplaced.Append(item);

}

}

if (tagReplaced.Length == 0)

{

Console.WriteLine(hTMLDoc);

}

else

{

Console.WriteLine(tagReplaced.ToString());

}

//Console.WriteLine();

//// other way

//string HTML = @"<p>Please visit <a href=""http://academy.telerik.com"">our site</a> to choose a training course. Also visit <a href=""www.devbg.org"">our forum</a> to discuss the courses.</p>";

//Console.WriteLine(Regex.Replace(HTML, @"<a href=""(.\*?)"">(.\*?)</a>", @"[URL=$1]$2[/URL]"));

}

}

/\* Problem 16. Date difference

Write a program that reads two dates in the format: day.month.year and calculates the number of days between them.

Example:

Enter the first date: 27.02.2006

Enter the second date: 3.03.2006

Distance: 4 days

\*/

// Note: Added file "Example Date Difference.txt" for faster testing

using System;

using System.Globalization; // needed for CultureInfo

class DateDifference

{

static void Main()

{

Console.Write("Enter the first date in the format [d.MM.yyyy]: ");

string startDate = Console.ReadLine();

Console.Write("Enter the second date in the format [d.MM.yyyy]: ");

string endDate = Console.ReadLine();

DateTime start = DateTime.ParseExact(startDate, "d.M.yyyy", CultureInfo.InvariantCulture);

DateTime end = DateTime.ParseExact(endDate, "d.M.yyyy", CultureInfo.InvariantCulture);

Console.WriteLine("\nDistance between {0} and {1} is {2} days\n", startDate, endDate, Math.Abs((end - start).TotalDays));

//// other way

//string[] Date1 = startDate.Split('.');

//string[] Date2 = endDate.Split('.');

//// DateTime.DateTime(int year, int month, int day)

//DateTime DateStart = new DateTime(int.Parse(Date1[2]), int.Parse(Date1[1]), int.Parse(Date1[0]));

//DateTime DateEnd = new DateTime(int.Parse(Date2[2]), int.Parse(Date2[1]), int.Parse(Date2[0]));

//int daysBetween = Math.Abs((int)(DateEnd - DateStart).TotalDays);

//Console.WriteLine("Distance: {0} days\n", daysBetween);

}

}

/\* Problem 17. Date in Bulgarian

Write a program that reads a date and time given in the format: day.month.year hour:minute:second

and prints the date and time after 6 hours and 30 minutes (in the same format) along with the day of week in Bulgarian.

\*/

using System;

using System.Globalization; // needed for CultureInfo

using System.Threading; // needed for Thread

class DateInBulgarian

{

static void Main()

{

// for easier testing

string input = "01.03.2015 00:00:00";

//// the real input

//Console.WriteLine("Enter a date and time in the format [day.month.year hour:minute:second]: ");

//string input = Console.ReadLine();

DateTime dateAndTime = DateTime.ParseExact(input, "dd.MM.yyyy HH:mm:ss", CultureInfo.InvariantCulture);

dateAndTime = dateAndTime.AddHours(6.5); // adding 6 hours and 30 minutes <= DateTime DateTime.AddHours(double value)

//dateAndTime = dateAndTime.AddHours(6).AddMinutes(30); // other way

Console.WriteLine("{0} {1}", dateAndTime.ToString("dddd", new CultureInfo("bg-BG")), dateAndTime);

// dddd => formatting string for dates => return day of Week as word

}

}

/\* Problem 18. Extract e-mails

Write a program for extracting all email addresses from given text.

All sub-strings that match the format <identifier>@<host>...<domain> should be recognized as emails.

\*/

/\* Note: Added input for faster testing

Input:

Please contact us by phone (+001 222 222 222) or by email at example@gmail.com or at test.user@yahoo.co.uk. This is not email: test@test. This also: @gmail.com. Neither this: a@a.b.

Output:

example@gmail.com

test.user@yahoo.co.uk

\*/

using System;

using System.Text.RegularExpressions; // needed for Regex

class Program

{

static void Main()

{

//// the real input

//string text = Console.ReadLine();

// input for faster testing

string text = "Please contact us by phone (+001 222 222 222) or by email at example@gmail.com or at test.user@yahoo.co.uk. This is not email: test@test. This also: @gmail.com. Neither this: a@a.b.";

string[] emails = text.Split(new[] { " ", ";", ",", ". " }, StringSplitOptions.RemoveEmptyEntries);

string[] validEmails = Array.FindAll(emails, IsValidEmail);

PrintEmails(validEmails);

}

static bool IsValidEmail(string email)

{

const string pattern = @"^([a-zA-Z0-9\_\-\.]+)@((\[[0-9]{1,3}" +

@"\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([a-zA-Z0-9\-]+\" +

@".)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$";

return new Regex(pattern).IsMatch(email);

}

static void PrintEmails(string[] validEmails)

{

Console.WriteLine("Extracted e-mail addresses from the sample text: ");

foreach (string email in validEmails)

{

Console.WriteLine(email);

}

Console.WriteLine();

}

}

/\* Problem 19. Dates from text in Canada

Write a program that extracts from a given text all dates that match the format DD.MM.YYYY.

Display them in the standard date format for Canada.

\*/

/\* Note: Added input for faster testing

Input:

I was born at 14.06.1980. My sister was born at 3.7.1984. In 5/1999 I graduated my high school. The law says (see section 7.3.12) that we are allowed to do this (section 7.4.2.9).

Output:

14.06.1980

3.7.1984

\*/

using System;

using System.Globalization; // needed for Regex

using System.Text.RegularExpressions; // needed for CultureInfo and DateTimeStyles

class DatesFromTextInCanada

{

static void Main()

{

//// the real input

//string text = Console.ReadLine();

// for faster testing

string text = "I was born at 14.06.1980. My sister was born at 3.7.1984. In 5/1999 I graduated my high school. The law says (see section 7.3.12) that we are allowed to do this (section 7.4.2.9).";

Console.WriteLine("Extracted dates from the sample text: ");

foreach (Match item in Regex.Matches(text, @"\b[0-9]{1,2}.[0-9]{1,2}.[0-9]{2,4}"))

{

DateTime date;

if (DateTime.TryParseExact(item.Value, "d.M.yyyy", CultureInfo.InvariantCulture, DateTimeStyles.None, out date))

{

Console.WriteLine(date.ToString(CultureInfo.GetCultureInfo("en-CA").DateTimeFormat.ShortDatePattern));

}

}

Console.WriteLine();

}

}

/\* Problem 20. Palindromes

Write a program that extracts from a given text all palindromes, e.g. ABBA, lamal, exe.

\*/

using System;

using System.Text;

class Palindromes

{

static void Main()

{

//// the real input

//string text = Console.ReadLine();

// the input for faster testing

string text = "Lamal? Every time I see a palindrome, I listen ABBA and feel like running an exe (lamal).";

var sb = new StringBuilder();

foreach (char ch in text)

{

if (char.IsPunctuation(ch))

{

sb.Append(ch);

}

}

sb.Append((char)32); // adding spaces

string punctuation = sb.ToString();

string[] words = text.Split(punctuation.ToCharArray(), StringSplitOptions.RemoveEmptyEntries);

Console.WriteLine("Palindromes: ");

foreach (string word in words)

{

bool isPalindrome = true;

for (int i = 0; i < word.Length / 2; i++)

{

// Lamal, lamaL are palindromes. => case insensitive

if (word[i] != word[word.Length - 1 - i] && word[i] != (char)(word[word.Length - 1 - i] - 32))

//if (word[i] != word[word.Length - 1 - i]) // Lamal, lamaL are not palindromes. => case sensitive

{

isPalindrome = false;

break;

}

}

if (isPalindrome && word.Length > 1) // don't need 1 letter words

{

Console.WriteLine(word);

}

}

}

}

/\* Problem 21. Letters count

Write a program that reads a string from the console and

prints all different letters in the string along with information how many times each letter is found.

\*/

using System;

using System.Collections.Generic; // needed for Dictionary

using System.Linq; // needed for OrderBy and OrderByDescending

class LettersCount

{

static void Main()

{

//// the real input

//string str = Console.ReadLine();

// input for faster testing

string str = "aaa bbb BBBB AAAAA bbb 0123456";

Dictionary<char, int> letters = new Dictionary<char, int>();

for (char i = 'a'; i <= 'z'; i++)

{

// void Dictionary<char,int>.Add(char key, int value) => adds the specified key and value to the dictionary

letters.Add(i, 0);

}

for (char i = 'A'; i <= 'Z'; i++)

{

letters.Add(i, 0);

}

foreach (char ch in str)

{

// bool ContainsKey(K) => check if there is an ordered pair with this key in the dictionary

if (letters.ContainsKey(ch))

{

letters[ch]++;

}

}

var orderedLetters = letters; // without additional sorting => a,b,...A,B,...

//a -> 3

//b -> 6

//A -> 5

//B -> 4

//var orderedLetters = letters.OrderBy(x => x.Key); //sort by the letter => A,B,..a,b..

//A -> 5

//B -> 4

//a -> 3

//b -> 6

//var orderedLetters = letters.OrderByDescending(x => x.Key); //sort by the letter in descending order => ..b,a...B,A

//b -> 6

//a -> 3

//B -> 4

//A -> 5

foreach (var letter in orderedLetters)

{

if (letter.Value != 0)

{

Console.WriteLine("{0} -> {1} ", letter.Key, letter.Value);

}

}

}

}

/\* Problem 22. Words count

Write a program that reads a string from the console and

lists all different words in the string along with information how many times each word is found.

\*/

using System;

using System.Collections.Generic; // needed for Dictionary and List<string>

using System.Linq; // needed for OrderBy, OrderByDescending, Distinct() and Count()

class WordsCount

{

static void Main()

{

//// the real input

//string str = Console.ReadLine();

// input for faster testing

string str = "Telerik Academy Nindja Programming Nindja programming Telerik Nindja";

// first way - with Dictionary<string, int>

string[] onlyWords = str.Split(new char[] { ' ', ',', '.', '-', '!', ':' }, StringSplitOptions.RemoveEmptyEntries);

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

foreach (string word in onlyWords)

{

if (allWords.ContainsKey(word))

{

allWords[word]++;

}

else

{

allWords.Add(word, 1);

}

}

//var orderedWords = allWords; // in order of appearance

var orderedWords = allWords.OrderByDescending(x => x.Value); // ordered by times found (value) in descending order

//var orderedWords = allWords.OrderBy(x => x.Key); // ordered by keys

foreach (var word in orderedWords)

{

Console.WriteLine("{0} -> {1} ", word.Key, word.Value);

}

//// second way - with List<string>

//List<string> listOnlyWords = new List<string>(str.Split(new char[] { ' ', ',', '.', '-', '!', ':' }, StringSplitOptions.RemoveEmptyEntries));

//List<string> allWords = listOnlyWords.Distinct().OrderBy(c => c).ToList();

//foreach (string word in allWords)

//{

// Console.WriteLine("{0} -> {1}", word, listOnlyWords.Count(x => x == word));

//}

}

}

/\* Problem 23. Series of letters

Write a program that reads a string from the console and

replaces all series of consecutive identical letters with a single one.

Example:

input output

aaaaabbbbbcdddeeeedssaa abcdedsa

\*/

using System;

using System.Text; // needed for StringBuilder

class SeriesOfLetters

{

static void Main()

{

//// the real input

//string str = Console.ReadLine();

// input for farster testing

string str = "aaaaabbbbbcdddeeeedssaa";

StringBuilder answer = new StringBuilder();

answer.Append(str[0]);

for (int i = 1; i < str.Length; i++)

{

if (str[i] != str[i - 1])

{

answer.Append(str[i]);

}

}

Console.WriteLine(answer.ToString());

}

}

/\* Problem 24. Order words

Write a program that reads a list of words, separated by spaces and prints the list in an alphabetical order.

\*/

using System;

using System.Linq; // needed for OrderBy and Select

class OrderWords

{

static void Main()

{

Console.WriteLine("Enter a list of words, separated by spaces: ");

string[] words = Console.ReadLine().Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

// first way

var ordered = words.OrderBy(x => x);

Console.WriteLine("\nWords sorted in alphabetical order:");

foreach (var item in ordered)

{

Console.WriteLine(item);

}

//// second way

//Array.Sort(words);

//Console.WriteLine("\nWords sorted in alphabetical order:\n{0}\n",

// string.Join("\n", words.Select(x => string.Format("=> {0}", x))));

}

}

/\* Problem 25. Extract text from HTML => Regex

Write a program that extracts from given HTML file its title (if available), and its body text without the HTML tags.

Example input:

<html>

<head><title>News</title></head>

<body><p><a href="http://academy.telerik.com">Telerik

Academy</a>aims to provide free real-world practical

training for young people who want to turn into

skillful .NET software engineers.</p></body>

</html>

Output:

Title: News

Text: Telerik Academy aims to provide free real-world practical training for young people who want to turn into skillful .NET software engineers.

\*/

using System;

using System.Text.RegularExpressions; // needed for Regex

class ExtractTextFromHTMLRegex

{

static void Main()

{

string text = @"<html><head><title>News</title></head><body><p><a href=""http://academy.telerik.com"">Telerik Academy</a>aims to provide free real-world practical training for young people who want to turn into skillful .NET software engineers.</p></body></html>";

MatchCollection tags = Regex.Matches(text, @"((?<=^|>)[^><]+?(?=<|$))");

int count = 1;

foreach (Match tag in tags)

{

if (count == 1)

{

Console.WriteLine("Title: {0}", tag);

Console.Write("Text: ");

}

else

{

Console.Write(tag + " ");

}

count++;

}

Console.WriteLine(Environment.NewLine);

}

}

/\* Problem 25. Extract text from HTML

Write a program that extracts from given HTML file its title (if available), and its body text without the HTML tags.

Example input:

<html>

<head><title>News</title></head>

<body><p><a href="http://academy.telerik.com">Telerik

Academy</a>aims to provide free real-world practical

training for young people who want to turn into

skillful .NET software engineers.</p></body>

</html>

Output:

Title: News

Text: Telerik Academy aims to provide free real-world practical training for young people who want to turn into skillful .NET software engineers.

\*/

**// unfinished!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!**

using System;

using System.Text; // needed for Stringbuilder

class ExtractTextFromHTML

{

static void Main()

{

string hTML = @"<html>

<head><title>News</title></head>

<body><p><a href=""http://academy.telerik.com"">Telerik

Academy</a>aims to provide free real-world practical

training for young people who want to turn into

skilful .NET software engineers.</p></body>

</html>";

StringBuilder answer = new StringBuilder();

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

{

if (hTML[i] == '<')

{

while (hTML[i] != '>')

{

i++;

}

continue;

}

answer.Append(hTML[i]);

}

Console.WriteLine(answer.ToString().Trim());

Console.WriteLine();

}

}

# Homework: Exception Handling

/\* Problem 1. Square root

Write a program that reads an integer number and calculates and prints its square root.

If the number is invalid or negative, print "Invalid number".

In all cases finally print "Good bye".

Use try-catch-finally block.

\*/

using System;

class SquareRoot

{

static void Main()

{

try

{

Console.Write("Enter a positive integer number: ");

uint num = uint.Parse(Console.ReadLine());

Console.WriteLine("{0:F2}", Math.Sqrt(num));

}

catch

{

Console.WriteLine("Invalid number.");

}

finally

{

Console.WriteLine("Good bye.");

}

}

}

/\* Problem 2. Enter numbers

Write a method ReadNumber(int start, int end) that enters an integer number in a given range [start...end].

If an invalid number or non-number text is entered, the method should throw an exception.

Using this method write a program that enters 10 numbers: a1, a2, ... a10, such that 1 < a1 < ... < a10 < 100

\*/

using System;

class EnterNumbers

{

static int MIN = 1;

static int MAX = 100;

const int COUNT = 10;

static void Main()

{

Console.WriteLine("Enter 10 numbers, so that 1 < n1 < n2 < ... < n10 < 100");

try

{

for (int i = 0; i < COUNT; i++)

{

MIN = ReadInteger();

}

}

catch (FormatException exception)

{

Console.WriteLine(exception.Message);

}

catch (ArgumentOutOfRangeException exception)

{

Console.WriteLine(exception.Message);

}

finally

{

Console.WriteLine("Good bye.");

}

}

private static int ReadInteger()

{

Console.Write("Enter number in range [{0}...{1}]: ", MIN + 1, MAX - 1);

int number = int.Parse(Console.ReadLine());

if (number <= MIN || number >= MAX)

{

throw new ArgumentOutOfRangeException();

}

return number;

}

}

/\* Problem 3. Read file contents

Write a program that enters file name along with its full file path

(e.g. C:\WINDOWS\win.ini), reads its contents and prints it on the console.

Find in MSDN how to use System.IO.File.ReadAllText(…).

Be sure to catch all possible exceptions and print user-friendly error messages.

\*/

using System;

using System.IO;

class ReadFileContents

{

static void Main()

{

Console.Write("Enter the full path of the file you want to read: ");

string filePath = Console.ReadLine();

try

{

string fileContent = File.ReadAllText(filePath);

Console.WriteLine("The content of the file is: ");

Console.WriteLine(fileContent);

}

catch (FileNotFoundException)

{

Console.WriteLine("The file '{0}' was not found!", filePath);

}

catch (ArgumentNullException)

{

Console.WriteLine("No file path is given!");

}

catch (ArgumentException)

{

Console.WriteLine("The entered file path is not correct!");

}

catch (NotSupportedException)

{

Console.WriteLine("The chosen file is not supported!");

}

}

}

/\* Problem 4. Download file

Write a program that downloads a file from Internet (e.g. Ninja image http://telerikacademy.com/Content/Images/news-img01.png)

and stores it the current directory.

Find in Google how to download files in C#.

Be sure to catch all exceptions and to free any used resources in the finally block.

\*/

using System;

using System.Net;

class DownloadFile

{

static void Main()

{

Console.WriteLine("Do You allow access of the program to download a picture from the web?\n");

string userChoice = string.Empty;

do

{

Console.Write("Enter your choice ([Y] / [N]): ");

userChoice = Console.ReadLine();

}

while (userChoice.ToLower() != "n" && userChoice.ToLower() != "y");

if (userChoice.ToLower() == "n")

{

return;

}

using (WebClient webClient = new WebClient())

{

try

{

Console.WriteLine("\nStart downloading...");

webClient.DownloadFile("http://telerikacademy.com/Content/Images/news-img01.png", "../../Telerik-Ninja.png");

Console.WriteLine("\n-> Download successfully!");

}

catch (ArgumentException)

{

Console.Error.WriteLine("\n-> Error: You have entered an empty URL!");

}

catch (WebException)

{

Console.Error.WriteLine("\n-> Error: You have entered an invalid URL!");

}

catch (NotSupportedException)

{

Console.Error.WriteLine("\n-> Error: This method does not support simultaneous downloads!");

}

finally

{

Console.WriteLine("\nGoodbye!\n");

}

}

}

}

# Homework: Text Files

/\* Problem 1. Odd lines

Write a program that reads a text file and prints on the console its odd lines.

\*/

using System;

using System.IO;

using System.Text;

public class OddLines

{

public static void Main()

{

Console.OutputEncoding = Encoding.UTF8;

StreamReader reader = new StreamReader("../../TextFile.txt");

using (reader)

{

int lineNumber = 0;

string line = reader.ReadLine();

while (line != null)

{

lineNumber++;

if (lineNumber % 2 != 0)

{

Console.WriteLine("Line {0}: {1}", lineNumber, line);

}

line = reader.ReadLine();

}

}

}

}

/\* Problem 2. Concatenate text files

Write a program that concatenates two text files into another text file.

\*/

using System;

using System.IO;

using System.Text;

public class ConcatenateTextFiles

{

public static void Main()

{

Console.BufferHeight = 300;

Console.OutputEncoding = Encoding.UTF8;

using (var output = File.Create("../../Result.txt"))

{

foreach (var file in new[] { "../../TextFileOne.txt",

"../../TextFileTwo.txt" })

{

using (var input = File.OpenRead(file))

{

input.CopyTo(output);

}

}

}

StreamReader reader = new StreamReader("../../Result.txt",

Encoding.GetEncoding("windows-1251"));

using (reader)

{

int lineNumber = 0;

string line = reader.ReadLine();

while (line != null)

{

lineNumber++;

Console.WriteLine(line);

line = reader.ReadLine();

}

}

}

}

/\* Problem 3. Line numbers

Write a program that reads a text file and inserts line numbers in front of each of its lines.

The result should be written to another text file.

\*/

using System;

using System.IO;

using System.Text;

public class LineNumbers

{

public static void Main()

{

Console.BufferHeight = 200;

Console.OutputEncoding = Encoding.UTF8;

using (StreamReader reader = new StreamReader("../../OdaZaLewski.txt", Encoding.GetEncoding("windows-1251")))

{

int lineNumber = 0;

string line = reader.ReadLine();

using (StreamWriter writer = new StreamWriter("../../Result.txt", false, Encoding.GetEncoding("windows-1251")))

{

while (line != null)

{

if (!String.IsNullOrWhiteSpace(line))

{

writer.WriteLine("{0} : {1}", lineNumber, line);

lineNumber++;

}

line = reader.ReadLine();

}

}

}

using (StreamReader readerResult = new StreamReader("../../Result.txt", Encoding.GetEncoding("windows-1251")))

{

int lineNumber = 0;

string line = readerResult.ReadLine();

while (line != null)

{

lineNumber++;

Console.WriteLine(line);

line = readerResult.ReadLine();

}

}

}

}

/\* Problem 4. Compare text files

Write a program that compares two text files line by line and

prints the number of lines that are the same and the number of lines that are different.

Assume the files have equal number of lines.

\*/

using System;

using System.IO;

using System.Text;

public class CompareTextFiles

{

public static void Main()

{

string[] linesOne = File.ReadAllLines("../../TextFileOne.txt");

string[] linesTwo = File.ReadAllLines("../../TextFileTwo.txt");

int maxLength = Math.Max(linesOne.Length, linesTwo.Length);

int minLength = Math.Min(linesOne.Length, linesTwo.Length);

for (int line = 0; line < maxLength; line++)

{

if (line < minLength)

{

if (linesOne[line].Equals(linesTwo[line]))

{

Console.WriteLine("At line {0} : lines from both the file are same.", line);

}

else

{

Console.WriteLine("At line {0} : lines are not same.", line);

}

}

else

{

Console.WriteLine("Line {0} : doesnt exits in {1} file.", line,

linesOne.Length == minLength ? "first" : "second");

}

}

}

}

/\* Problem 5. Maximal area sum

Write a program that reads a text file containing a square matrix of numbers.

Find an area of size 2 x 2 in the matrix, with a maximal sum of its elements.

The first line in the input file contains the size of matrix N.

Each of the next N lines contain N numbers separated by space.

The output should be a single number in a separate text file.

Example:

input output

4

2 3 3 4

0 2 3 4 17

3 7 1 2

4 3 3 2

\*/

using System;

using System.IO;

using System.Text;

public class MaximalAreaSum

{

private static int result = 0;

private static int[,] subMatrix = new int[2, 2];

public static void Main()

{

int[,] matrix;

int length;

using (StreamReader readerResult = new StreamReader("../../SquareMatrixOfNumbers.txt"))

{

string line = readerResult.ReadLine();

length = int.Parse(line);

matrix = new int[length, length];

char[] characters = new[] { ' ' };

for (int lineNumber = 0; lineNumber < length; lineNumber++)

{

line = readerResult.ReadLine();

string[] numbers = line.Split(characters, StringSplitOptions.RemoveEmptyEntries);

for (int charNumber = 0; charNumber < length; charNumber++)

{

matrix[lineNumber, charNumber] = int.Parse(numbers[charNumber]);

}

}

}

for (int vertikal = 0; vertikal < length - 1; vertikal++)

{

for (int horizontal = 0; horizontal < length - 1; horizontal++)

{

SubMatrixSum(matrix, vertikal, horizontal);

}

}

PrintMatrix(matrix);

PrintMatrix(subMatrix);

using (StreamWriter writer = new StreamWriter("../../result.txt"))

{

writer.Write(result);

Console.WriteLine("\nMaximal area sum = {0}\n", result);

Console.WriteLine("result.txt - Done\n");

}

}

private static void SubMatrixSum(int[,] array, int row, int column)

{

int sum = 0;

for (int i = 0; i < 4; i++)

{

sum += array[row + (i % 2), column + (i / 2)];

}

if (sum > result)

{

for (int i = 0; i < 4; i++)

{

subMatrix[i % 2, i / 2] = array[row + (i % 2), column + (i / 2)];

}

result = sum;

}

}

private static void PrintMatrix(int[,] array)

{

string line = new string('-', (array.GetLength(1) \* 6));

for (int column = 0; column < array.GetLength(0); column++)

{

Console.WriteLine(line + "-");

Console.Write("|");

for (int row = 0; row < array.GetLength(1); row++)

{

if ((array[column, row]) < 10)

{

Console.Write(" ");

}

else if ((array[column, row]) < 100)

{

Console.Write(" ");

}

Console.Write(" {0} |", (array[column, row]));

}

Console.WriteLine();

}

Console.WriteLine(line + "-");

}

}

/\* Problem 6. Save sorted names

Write a program that reads a text file containing a list of strings,

sorts them and saves them to another text file.

Example:

input.txt output.txt

Ivan George

Peter Ivan

Maria Maria

George Peter

\*/

using System;

using System.Collections.Generic;

using System.IO;

public class SaveSortedNames

{

public static void Main()

{

SaveSortedList(GetList());

}

private static void SaveSortedList(List<String> listOfNames)

{

using (StreamWriter writer = new StreamWriter("../../output.txt"))

{

for (int index = 0; index < listOfNames.Count; index++)

{

writer.WriteLine(listOfNames[index]);

}

Console.WriteLine("output.txt - Done");

}

}

private static List<String> GetList()

{

using (StreamReader reader = new StreamReader("../../input.txt"))

{

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

int lineNumber = 0;

string line = reader.ReadLine();

char[] separators = new char[] { ' ', '|' };

while (line != null)

{

string[] pointCoords = line.Split(separators,

StringSplitOptions.RemoveEmptyEntries);

foreach (var item in pointCoords)

{

listOfNames.Add(item);

}

lineNumber++;

line = reader.ReadLine();

}

listOfNames.Sort();

return listOfNames;

}

}

}

/\* Problem 7. Replace sub-string

Write a program that replaces all occurrences of the sub-string start with the sub-string finish in a text file.

Ensure it will work with large files (e.g. 100 MB).

\*/

using System;

using System.IO;

public class ReplaceSubString

{

public static void Main()

{

using (StreamReader streamReader = new StreamReader("../../Text.txt"))

{

using (StreamWriter streamWriter = new StreamWriter("../../Result.txt"))

{

string line = streamReader.ReadLine();

while (line != null)

{

streamWriter.WriteLine(line.Replace("start", "finish"));

line = streamReader.ReadLine();

}

Console.WriteLine("Result.txt - Done");

}

}

}

}

/\* Problem 8. Replace whole word

Modify the solution of the previous problem to replace only whole words (not strings).

\*/

using System;

using System.IO;

using System.Text.RegularExpressions;

public class ReplaceWholeWord

{

public static void Main()

{

using (StreamReader streamReader = new StreamReader("../../Text.txt"))

{

using (StreamWriter streamWriter = new StreamWriter("../../Result.txt"))

{

string line = streamReader.ReadLine();

while (line != null)

{

streamWriter.WriteLine(Regex.Replace(line, @"\bstart\b", "finish"));

line = streamReader.ReadLine();

}

Console.WriteLine("Result.txt - Done");

}

}

}

}

/\* Problem 9. Delete odd lines

Write a program that deletes from given text file all odd lines.

The result should be in the same file.

\*/

using System;

using System.IO;

using System.Text;

public class DeleteOddLines

{

public static void Main()

{

Console.OutputEncoding = Encoding.UTF8;

string[] lines = File.ReadAllLines("../../TextFileOriginal.txt",

Encoding.GetEncoding("windows-1251"));

for (int index = 0; index < lines.Length; index++)

{

Console.WriteLine(lines[index]);

}

using (StreamWriter streamWriter = new StreamWriter("../../TextFileResult.txt",

false, Encoding.GetEncoding("windows-1251")))

{

for (int index = 0; index < lines.Length; index++)

{

if (index % 2 == 0)

{

streamWriter.WriteLine(lines[index]);

}

}

Console.WriteLine("TextFileResult.txt - Done");

}

}

}

/\* Problem 10. Extract text from XML

Write a program that extracts from given XML file all the text without the tags.

Example:

<?xml version="1.0"><student><name>Pesho</name><age>21</age><interests count="3"><interest>Games</interest><interest>C#</interest><interest>Java</interest></interests></student>

\*/

using System;

using System.IO;

using System.Text;

using System.Text.RegularExpressions;

public class ExtractTextFromXML

{

public static void Main()

{

Console.WriteLine("================================================");

using (StreamReader streamReader = new StreamReader("../../TestXML.xml"))

{

for (int index; (index = streamReader.Read()) != -1; )

{

if (index == '>')

{

StringBuilder text = new StringBuilder();

while (((index = streamReader.Read()) != '<') && index != -1)

{

text.Append((char)index);

}

string newText = Convert.ToString(text).Trim();

if (!String.IsNullOrWhiteSpace(newText))

{

Console.WriteLine(newText);

}

}

}

}

Console.WriteLine("================================================");

string textXML = File.ReadAllText("../../TestXML.xml");

string pattern = @"(?s)>\s\*(.\*?)\s\*<";

var str = Regex.Matches(textXML, pattern);

for (int index = 0; index < str.Count; index++)

{

var result = str[index].Groups[1].Value;

if (!string.IsNullOrWhiteSpace(result))

{

Console.WriteLine(result);

}

}

Console.WriteLine("================================================");

}

}

/\* Problem 11. Prefix "test"

Write a program that deletes from a text file all words that start with the prefix test.

Words contain only the symbols 0...9, a...z, A...Z, \_.

\*/

using System;

using System.IO;

using System.Collections.Generic;

class DeleteAllWordsStartingWithTest

{

static void Main()

{

StreamReader reader = new StreamReader(@"..\..\Text.txt");

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

string currLine = reader.ReadLine();

while (currLine != null)

{

int index = currLine.IndexOf("Test");

if (index != -1)

{

if ((currLine[index + 4] != ' ') && (currLine[index + 4] != '.') && (currLine[index + 4] != ',') && (currLine[index + 4] != '-'))

{

currLine = currLine.Substring(0, index) + currLine.Substring(index + 4);

}

}

index = -1;

index = currLine.IndexOf("test");

if (index != -1)

{

if ((currLine[index + 4] != ' ') && (currLine[index + 4] != '.') && (currLine[index + 4] != ',') && (currLine[index + 4] != '-'))

{

currLine = currLine.Substring(0, index) + currLine.Substring(index + 4);

}

}

answer.Add(currLine);

currLine = reader.ReadLine();

}

reader.Close();

StreamWriter writer = new StreamWriter(@"..\..\Text.txt", false);

foreach (string item in answer)

{

writer.WriteLine(item);

}

writer.Close();

Console.WriteLine("All words that start with the prefix test deleted!");

}

}

/\* Problem 12. Remove words

Write a program that removes from a text file all words listed in given another text file.

Handle all possible exceptions in your methods.

\*/

using System;

using System.IO;

using System.Security;

using System.Text.RegularExpressions;

public class RemoveWords

{

public static void Main()

{

try

{

string listedWords = @"\b" + String.Join("|",

File.ReadAllLines("../../ListedWords.txt")) + @"\b";

using (StreamReader streamReader = new StreamReader("../../Text.txt"))

{

using (StreamWriter streamWriter = new StreamWriter("../../Result.txt"))

{

string line = streamReader.ReadLine();

while (line != null)

{

streamWriter.WriteLine(Regex.Replace(line,

listedWords, String.Empty));

line = streamReader.ReadLine();

}

Console.WriteLine("Result.txt - Done !");

}

}

}

catch (FileNotFoundException e)

{

Console.WriteLine(e.Message);

}

catch (DirectoryNotFoundException e)

{

Console.WriteLine(e.Message);

}

catch (IOException e)

{

Console.WriteLine(e.Message);

}

catch (SecurityException e)

{

Console.WriteLine(e.Message);

}

catch (UnauthorizedAccessException e)

{

Console.WriteLine(e.Message);

}

}

}

/\* Problem 13. Count words

Write a program that reads a list of words from the file words.txt and

finds how many times each of the words is contained in another file test.txt.

The result should be written in the file result.txt and

the words should be sorted by the number of their occurrences in descending order.

Handle all possible exceptions in your methods.

\*/

using System;

using System.IO;

using System.Security;

using System.Text;

using System.Text.RegularExpressions;

public class CountWords

{

public static void Main()

{

try

{

string[] listedWords = File.ReadAllLines("../../words.txt",

Encoding.GetEncoding("windows-1251"));

int[] counter = new int[listedWords.Length];

using (StreamReader reader = new StreamReader("../../test.txt",

Encoding.GetEncoding("windows-1251")))

{

int lineNumber = 0;

string line = reader.ReadLine();

while (line != null)

{

lineNumber++;

for (int index = 0; index < listedWords.Length; index++)

{

counter[index] += ((Regex

.Matches(line, @"\b" + listedWords[index] + @"\b").Count));

}

line = reader.ReadLine();

}

}

Array.Sort(counter, listedWords);

using (StreamWriter writer = new StreamWriter("../../result.txt"))

{

for (int index = listedWords.Length - 1; index >= 0; index--)

{

writer.WriteLine("{0}: {1}", listedWords[index], counter[index]);

}

Console.WriteLine("result.txt - Done !");

}

}

catch (FileNotFoundException e)

{

Console.WriteLine(e.Message);

}

catch (DirectoryNotFoundException e)

{

Console.WriteLine(e.Message);

}

catch (IOException e)

{

Console.WriteLine(e.Message);

}

catch (SecurityException e)

{

Console.WriteLine(e.Message);

}

catch (UnauthorizedAccessException e)

{

Console.WriteLine(e.Message);

}

}

}