Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

## «Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ)

Кафедра информационных систем и программной инженерии

# Лабораторная работа № 4

**по дисциплине**

**«Технологии программирования»**

Выполнил:

ст. гр. ИСТ-121

И.А. Большев

Принял:

В.В. Данилов

Владимир, 2023

Тема: Разработка Unit-тестов с использованием framework Nunit

Цель:

Изучить возможности Unit тестирование .NET приложением framework NUnit.

**Ход работы:**

Создадим проект консольного приложения. Создадим в нём 3 файла с классами Program.cs в котором будет ввод и вывод данных пользователю, Logic.cs в котором будет расположены методы обработки данных и Testing.cs в котором будут тесты для методов.

1. Метод, который выводит сокращённое названия дня недели по его номеру и в зависимости от языка выводит на русском или на английском

Напишем тест для этого метода.

[Test]

public void Test1()

{

LogicWork l = new LogicWork();

int lang = 0;

Assert.AreEqual("Пн", l.GetNameByDay(lang,1));

Assert.AreEqual("Вт", l.GetNameByDay(lang, 2));

Assert.AreEqual("Ср", l.GetNameByDay(lang, 3));

Assert.AreEqual("Чт", l.GetNameByDay(lang, 4));

Assert.AreEqual("Пт", l.GetNameByDay(lang, 5));

Assert.AreEqual("Сб", l.GetNameByDay(lang, 6));

Assert.AreEqual("Вс", l.GetNameByDay(lang, 7));

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 8));

lang = 1;

Assert.AreEqual("Mon", l.GetNameByDay(lang, 1));

Assert.AreEqual("Tue", l.GetNameByDay(lang, 2));

Assert.AreEqual("Wed", l.GetNameByDay(lang, 3));

Assert.AreEqual("Thu", l.GetNameByDay(lang, 4));

Assert.AreEqual("Fri", l.GetNameByDay(lang, 5));

Assert.AreEqual("Sat", l.GetNameByDay(lang, 6));

Assert.AreEqual("Sun", l.GetNameByDay(lang, 7));

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 8));

lang = 2;

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 1));

}

Теперь напишем метод, который обеспечит выполнение данного теста.

private static String[] Days\_ru = {"Пн", "Вт", "Ср", "Чт", "Пт", "Сб", "Вс" };

private static String[] Days\_en = { "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };

public String GetNameByDay(int lang, int number\_day)

{

String res = "";

String[] Days;

switch (lang)

{

case 0: Days = Days\_ru; break;

case 1: Days = Days\_en; break;

default: throw new ArgumentException("lang");

}

switch (number\_day)

{

case 1:

res = Days[0];

break;

case 2:

res = Days[1];

break;

case 3:

res = Days[2];

break;

case 4:

res = Days[3];

break;

case 5:

res = Days[4];

break;

case 6:

res = Days[5];

break;

case 7:

res = Days[6];

break;

}

if (res == "")

{

throw new ArgumentException("Такого дня недели не существует");

}

else

{

return res;

}

}

Запустим выполнение теста.

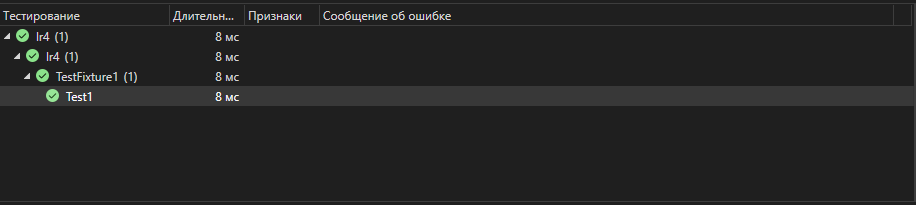


Рисунок 1 Скриншот выполнения теста

Выполним рефакторинг кода.

private static String[] Days\_ru = {"Пн", "Вт", "Ср", "Чт", "Пт", "Сб", "Вс" };

private static String[] Days\_en = { "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };

public String GetNameByDay(int lang, int number\_day)

{

String[] Days;

switch (lang)

{

case 0: Days = Days\_ru; break;

case 1: Days = Days\_en; break;

default: throw new ArgumentException("lang");

}

if (number\_day >= 1 && number\_day <= 7 ) {

return Days[number\_day - 1];

}

else

{

throw new ArgumentException("Такого дня недели не существует");

}

}

Проверим наш метод при помощи теста.

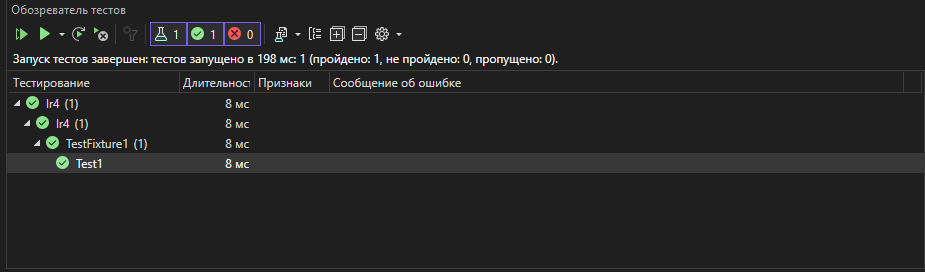


Рисунок 2 Скриншот выполнения теста

1. Метод, который считает гипотенузу по формуле Пифагора. На вход поступает 2 значения.

Напишем тест для этого метода.

[Test]

public void Test2()

{

LogicWork l = new LogicWork();

int[] expectedValue = { 5, 233, 41, 97 };

int[] kat1 = { 3, 105, 9, 65 };

int[] kat2 = {4, 208, 40, 72};

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

Assert.AreEqual(expectedValue[i], l.FindGipotenusa(kat1[i], kat2[i]));

}

}

Теперь напишем метод, который обеспечит выполнение данного теста.

public double FindGipotenusa(int par1, int par2)

{

double res = Math.Sqrt(par1 \* par1 + par2 \* par2);

return res;

}

Запустим выполнение теста.

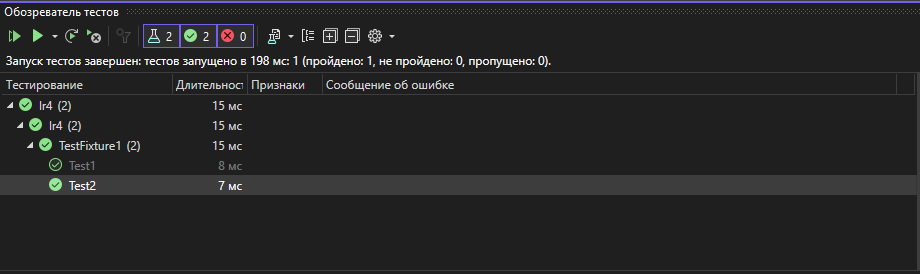


Рисунок 3 Скриншот выполнения теста

1. Метод, который будет выводить приветствие в зависимости от времени, а также в зависимости от языка будет выводить на русском или на английском.

Напишем тест для этого метода.

[Test]

public void Test3()

{

LogicWork l = new LogicWork();

String name = "Dima";

int lang = 0;

Assert.AreEqual("Доброе утро, Dima", l.NameTime(lang, name, "07:19"));

Assert.AreEqual("Добрый день, Dima", l.NameTime(lang, name, "14:32"));

Assert.AreEqual("Добрый вечер, Dima", l.NameTime(lang, name, "20:54"));

Assert.AreEqual("Доброй ночи, Dima", l.NameTime(lang, name, "00:12"));

lang = 1;

Assert.AreEqual("Good morning, Dima", l.NameTime(lang, name, "07:19"));

Assert.AreEqual("Good afternoon, Dima", l.NameTime(lang, name, "14:32"));

Assert.AreEqual("Good evening, Dima", l.NameTime(lang, name, "20:54"));

Assert.AreEqual("Good night, Dima", l.NameTime(lang, name, "00:12"));

lang = 2;

Assert.Throws<ArgumentException>(() => l.NameTime(lang, name, "00:12"));

}

Теперь напишем метод, который обеспечит выполнение данного теста.

private String[] Greatings\_ru = { "Доброе утро, ", "Добрый день, ", "Добрый вечер, ", "Доброй ночи, " };

private String[] Greatings\_en = { "Good morning, ", "Good afternoon, ", "Good evening, ", "Good night, " };

public String NameTime(int lang, String name, String time)

{

String[] Array;

int hour = int.Parse (time.Substring (0,2));

switch (lang)

{

case 0: Array = Greatings\_ru; break;

case 1: Array = Greatings\_en; break;

default: throw new ArgumentException("Array");

}

if (hour >= 5 && hour <= 11) {

return Array[0] + name;

}

else if (hour >= 12 && hour <= 16)

{

return Array[1] + name;

}

else if (hour >= 17 && hour <= 23)

{

return Array[2] + name;

}

else if (hour >= 0 && hour <= 4)

{

return Array[3] + name;

}

else

{

throw new ArgumentException("hour");

}

}

Запустим выполнение теста.

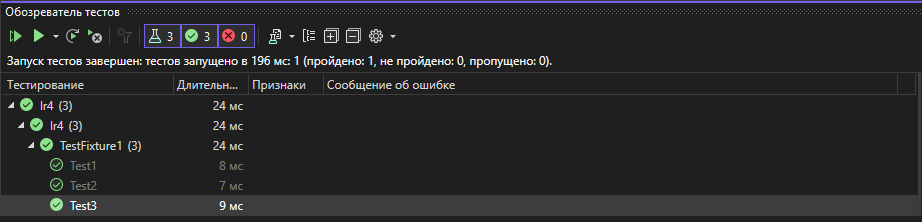


Рисунок 4 Скриншот выполнения теста

1. Метод, который будет выводить день и название месяца по дате, а также в зависимости от языка вывод будет осуществляться на русском или на английском языке.

Напишем тест для этого метода.

[Test]

public void Test4()

{

LogicWork l = new LogicWork();

int lang = 0;

Assert.AreEqual("25 Января", l.DatePrint(lang, "25.01.2010"));

Assert.AreEqual("18 Февраля", l.DatePrint(lang, "18.02.2023"));

Assert.AreEqual("1 Марта", l.DatePrint(lang, "01.03.2016"));

Assert.AreEqual("30 Апреля", l.DatePrint(lang, "30.04.2022"));

Assert.AreEqual("17 Мая", l.DatePrint(lang, "17.05.2018"));

Assert.AreEqual("24 Июня", l.DatePrint(lang, "24.06.2017"));

Assert.AreEqual("7 Июля", l.DatePrint(lang, "07.07.2018"));

Assert.AreEqual("12 Августа", l.DatePrint(lang, "12.08.2021"));

Assert.AreEqual("21 Сентября", l.DatePrint(lang, "21.09.2016"));

Assert.AreEqual("13 Октября", l.DatePrint(lang, "13.10.2022"));

Assert.AreEqual("9 Ноября", l.DatePrint(lang, "09.11.2014"));

Assert.AreEqual("31 Декабря", l.DatePrint(lang, "31.12.2020"));

lang = 1;

Assert.AreEqual("25 January", l.DatePrint(lang, "25.01.2010"));

Assert.AreEqual("18 February", l.DatePrint(lang, "18.02.2023"));

Assert.AreEqual("1 March", l.DatePrint(lang, "01.03.2016"));

Assert.AreEqual("30 April", l.DatePrint(lang, "30.04.2022"));

Assert.AreEqual("17 May", l.DatePrint(lang, "17.05.2018"));

Assert.AreEqual("24 June", l.DatePrint(lang, "24.06.2017"));

Assert.AreEqual("7 July", l.DatePrint(lang, "07.07.2018"));

Assert.AreEqual("12 August", l.DatePrint(lang, "12.08.2021"));

Assert.AreEqual("21 September", l.DatePrint(lang, "21.09.2016"));

Assert.AreEqual("13 October", l.DatePrint(lang, "13.10.2022"));

Assert.AreEqual("9 November", l.DatePrint(lang, "09.11.2014"));

Assert.AreEqual("31 December", l.DatePrint(lang, "31.12.2020"));

lang = 2;

Assert.Throws<ArgumentException>(() => l.DatePrint(lang, "31.12.2020"));

}

Теперь напишем метод, который обеспечит выполнение данного теста.

public String DatePrint(int lang, String date)

{

int day = int.Parse(date.Substring(0,2));

int monthNumber = int.Parse(date.Substring(3,2));

String res = Convert.ToString(day);

if (lang == 0)

{

switch (monthNumber)

{

case 1: res += " Января"; break;

case 2: res += " Февраля"; break;

case 3: res += " Марта"; break;

case 4: res += " Апреля"; break;

case 5: res += " Мая"; break;

case 6: res += " Июня"; break;

case 7: res += " Июля"; break;

case 8: res += " Августа"; break;

case 9: res += " Сентября"; break;

case 10: res += " Октября"; break;

case 11: res += " Ноября"; break;

case 12: res += " Декабря"; break;

}

}

else if (lang == 1)

{

switch (monthNumber)

{

case 1: res += " January"; break;

case 2: res += " February"; break;

case 3: res += " March"; break;

case 4: res += " April"; break;

case 5: res += " May"; break;

case 6: res += " June"; break;

case 7: res += " July"; break;

case 8: res += " August"; break;

case 9: res += " September"; break;

case 10: res += " October"; break;

case 11: res += " November"; break;

case 12: res += " December"; break;

}

}

else

{

throw new ArgumentException();

}

return res;

}

Запустим выполнение теста.

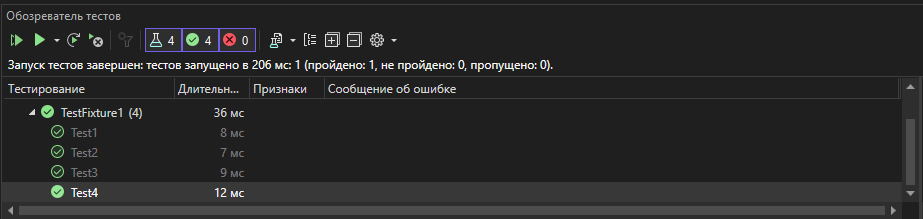


Рисунок 5 Скриншот выполнения теста

Выполним рефакторинг кода.

private String[] Mounts\_ru = { " Января", " Февраля", " Марта", " Апреля", " Мая", " Июня", " Июля", " Августа", " Сентября", " Октября", " Ноября", " Декабря" };

private String[] Mounts\_en = { " January", " February", " March", " April", " May", " June", " July", " August", " September", " October", " November", " December" };

public String DatePrint(int lang, String date)

{

int day = int.Parse(date.Substring(0,2));

int monthNumber = int.Parse(date.Substring(3,2));

String res = Convert.ToString(day);

switch (lang)

{

case 0: return res + Mounts\_ru[monthNumber - 1];

case 1: return res + Mounts\_en[monthNumber - 1];

default: throw new ArgumentException();

}

}

Проверим наш метод при помощи теста.

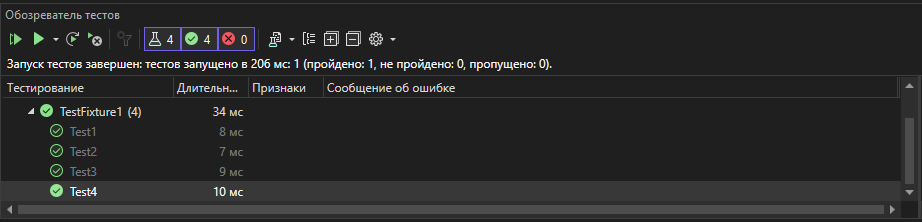


Рисунок 6 Скриншот выполнения теста

1. Метод, который будет выводить язык интерфейса, дату, название месяца, день недели и приветствие пользователя.

Напишем тест для этого метода.

[Test]

public void Test5()

{

LogicWork l = new LogicWork();

String name = "Dima";

int lang = 2;

Assert.Throws<ArgumentException>(() => l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

lang = 0;

Assert.AreEqual("Язык: Ru \n17 Мая Пн \nДобрый день, Dima", l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

Assert.AreEqual("Язык: Ru \n11 Февраля Вт \nДоброе утро, Dima", l.GreatingBuild(lang, name, "06:20", "11.02.2020", 2));

Assert.AreEqual("Язык: Ru \n20 Сентября Вс \nДобрый вечер, Dima", l.GreatingBuild(lang, name, "20:32", "20.09.2010", 7));

Assert.AreEqual("Язык: Ru \n9 Марта Пт \nДоброй ночи, Dima", l.GreatingBuild(lang, name, "02:32", "09.03.2008", 5));

Assert.AreEqual("Язык: Ru \n31 Декабря Сб \nДоброе утро, Dima", l.GreatingBuild(lang, name, "10:32", "31.12.2021", 6));

lang = 1;

Assert.AreEqual("Language: En \n17 May Mon \nGood afternoon, Dima", l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

Assert.AreEqual("Language: En \n11 February Tue \nGood morning, Dima", l.GreatingBuild(lang, name, "06:20", "11.02.2020", 2));

Assert.AreEqual("Language: En \n20 September Sun \nGood evening, Dima", l.GreatingBuild(lang, name, "20:32", "20.09.2010", 7));

Assert.AreEqual("Language: En \n9 March Fri \nGood night, Dima", l.GreatingBuild(lang, name, "02:32", "09.03.2008", 5));

Assert.AreEqual("Language: En \n31 December Sat \nGood morning, Dima", l.GreatingBuild(lang, name, "10:32", "31.12.2021", 6));

}

Теперь напишем метод, который обеспечит выполнение данного теста.

public String GreatingBuild(int lang, String name, String time, String date, int dayNumber)

{

String res;

switch (lang)

{

case 0: res = "Язык: Ru"; break;

case 1: res = "Language: En"; break;

default: throw new ArgumentException("lang");

}

res += " \n" + DatePrint(lang, date) + " " + GetNameByDay(lang, dayNumber) + " \n" + NameTime(lang, name, time);

return res;

}

Запустим выполнение теста.

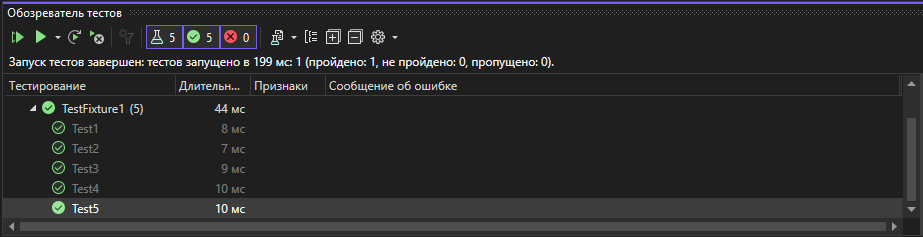


Рисунок 7 Скриншот выполнения теста

Теперь допишем в Program.cs считывание времени, даты и имени пользователя, а также ввод, вывод данных и вызовы необходимых методов. Затем запустим программу и проверим результат.

using lr4;

LogicWork l = new LogicWork();

DateTime now = DateTime.Now;

DayOfWeek dayOfWeek = now.DayOfWeek;

int dayOfWeekNumber = (int)dayOfWeek;

String nowDate = now.ToString("dd.MM.yyyy");

String nowTime = now.ToString("HH:mm:ss");

string username = Environment.UserName;

int language = 1;

Console.WriteLine(l.GreatingBuild(language, username, nowTime, nowDate, dayOfWeekNumber));

String line1 = "";

String line2 = "";

String line3 = "";

bool flag = true;

switch (language)

{

case (0):

line1 = "Введите первый катет:";

line2 = "Введите второй катет:";

line3 = "Гипотенуза = ";

break;

case (1):

line1 = "Enter the first leg:";

line2 = "Enter the second leg:";

line3 = "Hypotenuse = ";

break;

default:

flag = false;

break;

}

if (flag)

{

Console.Write(line1);

int kat1 = int.Parse(Console.ReadLine());

Console.Write(line2);

int kat2 = int.Parse(Console.ReadLine());

Console.WriteLine(line3 + l.FindGipotenusa(kat1, kat2).ToString("0.0000"));

}

else

{

Console.Write("ERROR");

}

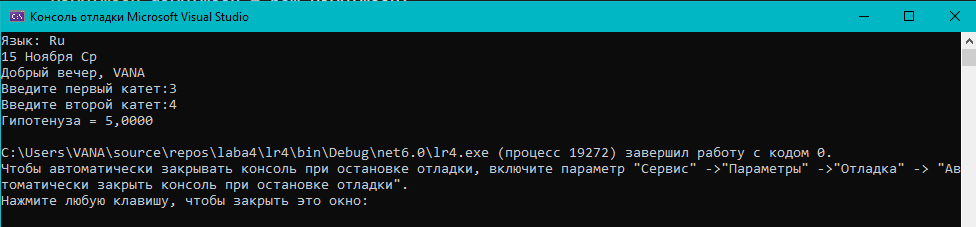


Рисунок 8 Скриншот выполнения программы

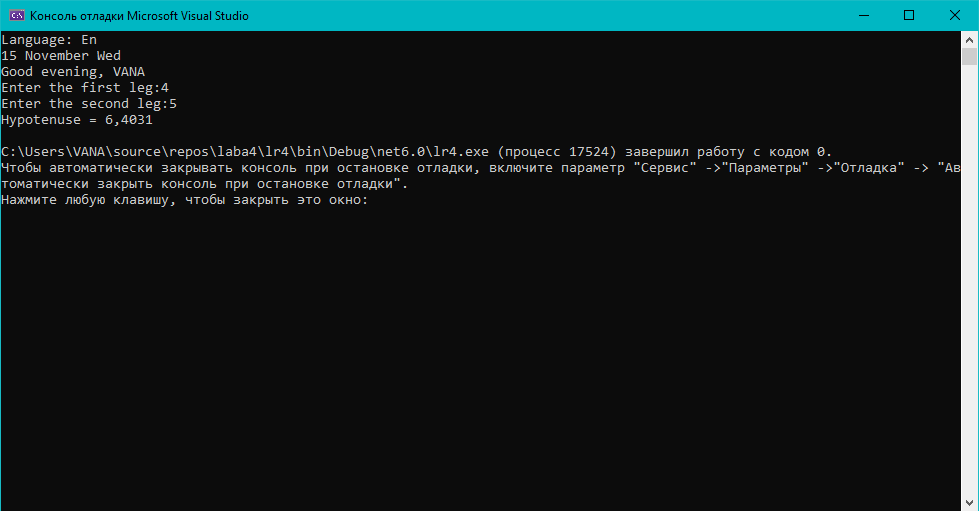


Рисунок 9 Скриншот выполнения программы

Содержимое файлов.

Листинг Testing.cs

using Microsoft.VisualStudio.TestPlatform.CommunicationUtilities;

using NUnit.Framework;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

namespace lr4

{

[TestFixture]

public class TestFixture1

{

[Test]

public void Test1()

{

LogicWork l = new LogicWork();

int lang = 0;

Assert.AreEqual("Пн", l.GetNameByDay(lang,1));

Assert.AreEqual("Вт", l.GetNameByDay(lang, 2));

Assert.AreEqual("Ср", l.GetNameByDay(lang, 3));

Assert.AreEqual("Чт", l.GetNameByDay(lang, 4));

Assert.AreEqual("Пт", l.GetNameByDay(lang, 5));

Assert.AreEqual("Сб", l.GetNameByDay(lang, 6));

Assert.AreEqual("Вс", l.GetNameByDay(lang, 7));

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 8));

lang = 1;

Assert.AreEqual("Mon", l.GetNameByDay(lang, 1));

Assert.AreEqual("Tue", l.GetNameByDay(lang, 2));

Assert.AreEqual("Wed", l.GetNameByDay(lang, 3));

Assert.AreEqual("Thu", l.GetNameByDay(lang, 4));

Assert.AreEqual("Fri", l.GetNameByDay(lang, 5));

Assert.AreEqual("Sat", l.GetNameByDay(lang, 6));

Assert.AreEqual("Sun", l.GetNameByDay(lang, 7));

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 8));

lang = 2;

Assert.Throws<ArgumentException>(() => l.GetNameByDay(lang, 1));

}

[Test]

public void Test2()

{

LogicWork l = new LogicWork();

int[] expectedValue = { 5, 233, 41, 97 };

int[] kat1 = { 3, 105, 9, 65 };

int[] kat2 = {4, 208, 40, 72};

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

Assert.AreEqual(expectedValue[i], l.FindGipotenusa(kat1[i], kat2[i]));

}

}

[Test]

public void Test3()

{

LogicWork l = new LogicWork();

String name = "Dima";

int lang = 0;

Assert.AreEqual("Доброе утро, Dima", l.NameTime(lang, name, "07:19"));

Assert.AreEqual("Добрый день, Dima", l.NameTime(lang, name, "14:32"));

Assert.AreEqual("Добрый вечер, Dima", l.NameTime(lang, name, "20:54"));

Assert.AreEqual("Доброй ночи, Dima", l.NameTime(lang, name, "00:12"));

lang = 1;

Assert.AreEqual("Good morning, Dima", l.NameTime(lang, name, "07:19"));

Assert.AreEqual("Good afternoon, Dima", l.NameTime(lang, name, "14:32"));

Assert.AreEqual("Good evening, Dima", l.NameTime(lang, name, "20:54"));

Assert.AreEqual("Good night, Dima", l.NameTime(lang, name, "00:12"));

lang = 2;

Assert.Throws<ArgumentException>(() => l.NameTime(lang, name, "00:12"));

}

[Test]

public void Test4()

{

LogicWork l = new LogicWork();

int lang = 0;

Assert.AreEqual("25 Января", l.DatePrint(lang, "25.01.2010"));

Assert.AreEqual("18 Февраля", l.DatePrint(lang, "18.02.2023"));

Assert.AreEqual("1 Марта", l.DatePrint(lang, "01.03.2016"));

Assert.AreEqual("30 Апреля", l.DatePrint(lang, "30.04.2022"));

Assert.AreEqual("17 Мая", l.DatePrint(lang, "17.05.2018"));

Assert.AreEqual("24 Июня", l.DatePrint(lang, "24.06.2017"));

Assert.AreEqual("7 Июля", l.DatePrint(lang, "07.07.2018"));

Assert.AreEqual("12 Августа", l.DatePrint(lang, "12.08.2021"));

Assert.AreEqual("21 Сентября", l.DatePrint(lang, "21.09.2016"));

Assert.AreEqual("13 Октября", l.DatePrint(lang, "13.10.2022"));

Assert.AreEqual("9 Ноября", l.DatePrint(lang, "09.11.2014"));

Assert.AreEqual("31 Декабря", l.DatePrint(lang, "31.12.2020"));

lang = 1;

Assert.AreEqual("25 January", l.DatePrint(lang, "25.01.2010"));

Assert.AreEqual("18 February", l.DatePrint(lang, "18.02.2023"));

Assert.AreEqual("1 March", l.DatePrint(lang, "01.03.2016"));

Assert.AreEqual("30 April", l.DatePrint(lang, "30.04.2022"));

Assert.AreEqual("17 May", l.DatePrint(lang, "17.05.2018"));

Assert.AreEqual("24 June", l.DatePrint(lang, "24.06.2017"));

Assert.AreEqual("7 July", l.DatePrint(lang, "07.07.2018"));

Assert.AreEqual("12 August", l.DatePrint(lang, "12.08.2021"));

Assert.AreEqual("21 September", l.DatePrint(lang, "21.09.2016"));

Assert.AreEqual("13 October", l.DatePrint(lang, "13.10.2022"));

Assert.AreEqual("9 November", l.DatePrint(lang, "09.11.2014"));

Assert.AreEqual("31 December", l.DatePrint(lang, "31.12.2020"));

lang = 2;

Assert.Throws<ArgumentException>(() => l.DatePrint(lang, "31.12.2020"));

}

[Test]

public void Test5()

{

LogicWork l = new LogicWork();

String name = "Dima";

int lang = 2;

Assert.Throws<ArgumentException>(() => l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

lang = 0;

Assert.AreEqual("Язык: Ru \n17 Мая Пн \nДобрый день, Dima", l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

Assert.AreEqual("Язык: Ru \n11 Февраля Вт \nДоброе утро, Dima", l.GreatingBuild(lang, name, "06:20", "11.02.2020", 2));

Assert.AreEqual("Язык: Ru \n20 Сентября Вс \nДобрый вечер, Dima", l.GreatingBuild(lang, name, "20:32", "20.09.2010", 7));

Assert.AreEqual("Язык: Ru \n9 Марта Пт \nДоброй ночи, Dima", l.GreatingBuild(lang, name, "02:32", "09.03.2008", 5));

Assert.AreEqual("Язык: Ru \n31 Декабря Сб \nДоброе утро, Dima", l.GreatingBuild(lang, name, "10:32", "31.12.2021", 6));

lang = 1;

Assert.AreEqual("Language: En \n17 May Mon \nGood afternoon, Dima", l.GreatingBuild(lang, name, "14:32", "17.05.2018", 1));

Assert.AreEqual("Language: En \n11 February Tue \nGood morning, Dima", l.GreatingBuild(lang, name, "06:20", "11.02.2020", 2));

Assert.AreEqual("Language: En \n20 September Sun \nGood evening, Dima", l.GreatingBuild(lang, name, "20:32", "20.09.2010", 7));

Assert.AreEqual("Language: En \n9 March Fri \nGood night, Dima", l.GreatingBuild(lang, name, "02:32", "09.03.2008", 5));

Assert.AreEqual("Language: En \n31 December Sat \nGood morning, Dima", l.GreatingBuild(lang, name, "10:32", "31.12.2021", 6));

}

}

}

Листинг lr4.csproj

<Project Sdk="Microsoft.NET.Sdk">

<PropertyGroup>

<OutputType>Exe</OutputType>

<TargetFramework>net6.0</TargetFramework>

<ImplicitUsings>enable</ImplicitUsings>

<Nullable>enable</Nullable>

</PropertyGroup>

<ItemGroup>

<PackageReference Include="Microsoft.NET.Test.Sdk" Version="17.8.0" />

<PackageReference Include="NUnit" Version="3.14.0" />

<PackageReference Include="NUnit3TestAdapter" Version="4.5.0" />

</ItemGroup>

</Project>

Листинг Logic.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lr4

{

public class LogicWork

{

private String[] Mounts\_ru = { " Января", " Февраля", " Марта", " Апреля", " Мая", " Июня", " Июля", " Августа", " Сентября", " Октября", " Ноября", " Декабря" };

private String[] Mounts\_en = { " January", " February", " March", " April", " May", " June", " July", " August", " September", " October", " November", " December" };

private String[] Greatings\_ru = { "Доброе утро, ", "Добрый день, ", "Добрый вечер, ", "Доброй ночи, " };

private String[] Greatings\_en = { "Good morning, ", "Good afternoon, ", "Good evening, ", "Good night, " };

private static String[] Days\_ru = {"Пн", "Вт", "Ср", "Чт", "Пт", "Сб", "Вс" };

private static String[] Days\_en = { "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun" };

public String GetNameByDay(int lang, int number\_day)

{

String[] Days;

switch (lang)

{

case 0: Days = Days\_ru; break;

case 1: Days = Days\_en; break;

default: throw new ArgumentException("lang");

}

if (number\_day >= 1 && number\_day <= 7 ) {

return Days[number\_day - 1];

}

else

{

throw new ArgumentException("Такого дня недели не существует");

}

}

public double FindGipotenusa(int par1, int par2)

{

double res = Math.Sqrt(par1 \* par1 + par2 \* par2);

return res;

}

public String DatePrint(int lang, String date)

{

int day = int.Parse(date.Substring(0,2));

int monthNumber = int.Parse(date.Substring(3,2));

String res = Convert.ToString(day);

switch (lang)

{

case 0: return res + Mounts\_ru[monthNumber - 1];

case 1: return res + Mounts\_en[monthNumber - 1];

default: throw new ArgumentException();

}

}

public String NameTime(int lang, String name, String time)

{

String[] Array;

int hour = int.Parse (time.Substring (0,2));

switch (lang)

{

case 0: Array = Greatings\_ru; break;

case 1: Array = Greatings\_en; break;

default: throw new ArgumentException("Array");

}

if (hour >= 5 && hour <= 11) {

return Array[0] + name;

}

else if (hour >= 12 && hour <= 16)

{

return Array[1] + name;

}

else if (hour >= 17 && hour <= 23)

{

return Array[2] + name;

}

else if (hour >= 0 && hour <= 4)

{

return Array[3] + name;

}

else

{

throw new ArgumentException("hour");

}

}

public String GreatingBuild(int lang, String name, String time, String date, int dayNumber)

{

String res;

switch (lang)

{

case 0: res = "Язык: Ru"; break;

case 1: res = "Language: En"; break;

default: throw new ArgumentException("lang");

}

res += " \n" + DatePrint(lang, date) + " " + GetNameByDay(lang, dayNumber) + " \n" + NameTime(lang, name, time);

return res;

}

}

}

Листинг Program.cs

using lr4;

LogicWork l = new LogicWork();

DateTime now = DateTime.Now;

DayOfWeek dayOfWeek = now.DayOfWeek;

int dayOfWeekNumber = (int)dayOfWeek;

String nowDate = now.ToString("dd.MM.yyyy");

String nowTime = now.ToString("HH:mm:ss");

string username = Environment.UserName;

int language = 1;

Console.WriteLine(l.GreatingBuild(language, username, nowTime, nowDate, dayOfWeekNumber));

String line1 = "";

String line2 = "";

String line3 = "";

bool flag = true;

switch (language)

{

case (0):

line1 = "Введите первый катет:";

line2 = "Введите второй катет:";

line3 = "Гипотенуза = ";

break;

case (1):

line1 = "Enter the first leg:";

line2 = "Enter the second leg:";

line3 = "Hypotenuse = ";

break;

default:

flag = false;

break;

}

if (flag)

{

Console.Write(line1);

int kat1 = int.Parse(Console.ReadLine());

Console.Write(line2);

int kat2 = int.Parse(Console.ReadLine());

Console.WriteLine(line3 + l.FindGipotenusa(kat1, kat2).ToString("0.0000"));

}

else

{

Console.Write("ERROR");

}

**Вывод**

Таким образом, в ходе выполнения работы были изучены возможности Unit тестирования .NET приложения framework NUnit.