Министерство науки и высшего образования Российской Федерации

**Муромский институт (филиал)**

федерального государственного бюджетного образовательного учреждения высшего образования

**«Владимирский государственный университет**

**Имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(МИВлГУ)**

Факультет ИТР

Кафедра ПИн

ЛАБОРАТОРНАЯ

РАБОТА №3

по Тестирование ПO

Тема Модульное тестирование на примере классов

Руководитель

Колпаков А. А.

(фамилия, инициалы)

(подпись) (дата)

Студент ПИн-121

(группа)

Ермилов М.В.

(фамилия, инициалы)

(подпись) (дата)

Муром 2024

**Тема: модульное тестирование на примере классов.**

**Цель работы:** изучить и освоить основные принципы построения модульных тестов на основе классов в Visual Studio.

**Задание на лабораторную работу:**

1. Изучить материал, предоставленный в данных методических указаниях к лабораторной работе;
2. Разработать класс Dates (использование класса Data Time запрещено!), который:
3. хранит в себе данные времени;
4. позволяет сравнивать время (операции сравнения между двумя экземплярами класса);
5. позволяет проводить математические операции с временем (арифметические операции между двумя экземплярами класса, умножение и деление на число);
6. добавлять / вычитать годы, месяцы, недели и т.д.;
7. выводить информацию в удобочитаемом виде (например «1 год 11 месяцев 28 дней 6 часов 3 минуты 56 секунд»);
8. началом отсчёта должен быть текущий год.
9. Разработать модульные тесты (минимум 5 тестов на каждый метод, также должны быть методы проверки исключений);
10. Провести тестирование разработанного класса;
11. Составить отчёт. В отчёте отобразить:

* цели и задачи лабораторной работы;
* личное задание, согласно варианту;
* исходный код приложения;

вывод по данной лабораторной работе.

**Program.cs**

using System;

namespace DataTimePersonal

{

class Program

{

static void Main(string[] args)

{

DataTimeLigic dt = new DataTimeLigic();

DataTimeLigic dt2 = new DataTimeLigic();

dt2.AddSeconds(1);

dt2.AddMinutes(1);

dt2.AddHours(1);

dt2.AddDays(30);

dt2.AddMonths(1);

dt2.AddYears(1);

Console.WriteLine($"Current date: {dt.GetDate()}\nAdd date: {dt2.GetDate()}");

Console.WriteLine("Разница между датами: " + dt.CompareDates(dt2));

Console.WriteLine($"Current date: {dt.GetDate()}");

dt.DivideBy(2);

Console.WriteLine($"Date /2: {dt.GetDate()}");

}

}

public class DataTimeLigic

{

private int years;

private int months;

private int days;

private int hours;

private int minutes;

private int seconds;

public DataTimeLigic()

{

DateTime now = DateTime.Now;

years = now.Year;

months = now.Month;

days = now.Day;

hours = now.Hour;

minutes = now.Minute;

seconds = now.Second;

}

public DataTimeLigic(DateTime date)

{

years = date.Year;

months = date.Month;

days = date.Day;

hours = date.Hour;

minutes = date.Minute;

seconds = date.Second;

}

public void SetDate(int newYears, int newMonths, int newDays)

{

years = newYears;

months = newMonths;

days = newDays;

}

public int GetYears()

{

return years;

}

public int GetMonths()

{

return months;

}

public int GetDays()

{

return days;

}

public int GetHours()

{

return hours;

}

public int GetMinutes()

{

return minutes;

}

public int GetSeconds()

{

return seconds;

}

public void AddYears(int value)

{

years += value;

}

public void AddMonths(int value)

{

if (months + value <= 12)

months += value;

else

{

years += (value + months) / 12;

months += (value + months) % 12;

}

}

public void AddDays(int value)

{

if (days + value <= 31)

days += value;

else

{

months += (value + days) / 31;

days += (value + days) % 31;

}

}

public void AddHours(int value)

{

if (hours + value <= 23)

hours += value;

else

{

days += (value + hours) / 23;

hours += (value + hours) % 23;

}

}

public void AddMinutes(int value)

{

if (minutes + value <= 60)

minutes += value;

else

{

hours += value / 60;

minutes += value % 60;

}

}

public void AddSeconds(int value)

{

if (seconds + value <= 60)

seconds += value;

else

{

minutes += value / 60;

seconds += value % 60;

}

}

public void SubtractYears(int value)

{

if (years - value >= 0)

years -= value;

else

years = 0;

}

public void SubtractMonths(int value)

{

if (months - value >= 0)

months -= value;

else

{

years -= value / 12;

months -= value % 12;

}

}

public void SubtractDays(int value)

{

if (days - value >= 0)

days -= value;

else

{

months -= value / 31;

days -= value % 31;

}

}

public void SubtractHours(int value)

{

if (hours - value >= 0)

hours -= value;

else

{

days -= value / 24;

hours -= value % 24;

}

}

public void SubtractMinutes(int value)

{

if (minutes - value >= 0)

minutes -= value;

else

{

hours -= value / 60;

minutes -= value % 60;

}

}

public void SubtractSeconds(int value)

{

if (seconds - value >= 0)

seconds -= value;

else

{

minutes -= value / 60;

seconds -= value % 60;

}

}

public void MultiplyBy(int factor)

{

years \*= factor;

months \*= factor;

days \*= factor;

hours \*= factor;

minutes \*= factor;

seconds \*= factor;

}

public void DivideBy(int divisor)

{

years /= divisor;

months /= divisor;

days /= divisor;

hours /= divisor;

minutes /= divisor;

seconds /= divisor;

}

public string GetDateTime()

{

return $"{years} год {months} месяцев {days} дней {hours} часов {minutes} минут {seconds} секунд";

}

public string GetDate()

{

return $"{years} год {months} месяцев {days} дней";

}

public string CompareDates(DataTimeLigic otherDate)

{

TimeSpan timeDifference = new TimeSpan(days, hours, minutes, seconds) -

new TimeSpan(otherDate.days, otherDate.hours, otherDate.minutes, otherDate.seconds);

int yearsDiff = Math.Abs(years - otherDate.years);

int monthsDiff = Math.Abs(months - otherDate.months);

int daysDiff = Math.Abs(timeDifference.Days % 30);

int hoursDiff = Math.Abs(timeDifference.Hours);

int minutesDiff = Math.Abs(timeDifference.Minutes);

int secondsDiff = Math.Abs(timeDifference.Seconds);

string difference = $"{yearsDiff} год {monthsDiff} месяцев {daysDiff} дней {hoursDiff} часов {minutesDiff} минут {secondsDiff} секунд";

return difference;

}

}

}

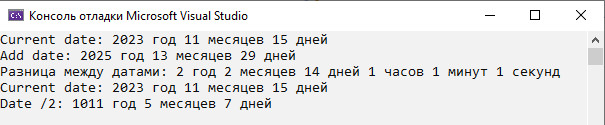


Рисунок 1 - Запуск консольного приложения

**UnitTest1.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

namespace TestProject1

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void FirstTest()

{

DataTimeLigic dt = new DataTimeLigic();

int expected = 2023;

int result = dt.GetYears();

Assert.AreEqual(expected, result);

}

[TestMethod]

public void AddYears\_UpdatesYears()

{

DataTimeLigic date = new DataTimeLigic();

date.AddYears(5);

Assert.AreEqual(2028, date.GetYears());

}

[TestMethod]

public void SubtractYears\_UpdatesYears()

{

DataTimeLigic date = new DataTimeLigic();

date.SubtractYears(3);

Assert.AreEqual(2020, date.GetYears());

}

[TestMethod]

public void AddMonths\_UpdatesMonths()

{

DataTimeLigic date = new DataTimeLigic();

date.AddMonths(8);

Assert.AreEqual(18, date.GetMonths());

}

[TestMethod]

public void SubtractMonths\_UpdatesMonths()

{

DataTimeLigic date = new DataTimeLigic();

date.SubtractMonths(4);

Assert.AreEqual(7, date.GetMonths());

}

[TestMethod]

public void SetDate\_UpdatesDate()

{

DataTimeLigic date = new DataTimeLigic();

date.SetDate(2023, 11, 1);

Assert.AreEqual(2023, date.GetYears());

Assert.AreEqual(11, date.GetMonths());

Assert.AreEqual(1, date.GetDays());

}

[TestMethod]

public void GetDate\_ReturnsDate()

{

DataTimeLigic date = new DataTimeLigic();

date.SetDate(2023, 11, 1);

string result = date.GetDate();

Assert.AreEqual("2023 год 11 месяцев 1 дней", result);

}

[TestMethod]

public void MultiplyBy\_MultipliesValues()

{

DataTimeLigic date = new DataTimeLigic();

date.MultiplyBy(2);

Assert.AreEqual(4046, date.GetYears());

}

[TestMethod]

public void DivideBy\_DividesValues()

{

DataTimeLigic date = new DataTimeLigic();

date.SetDate(2023, 12, 15);

date.DivideBy(3);

Assert.AreEqual(674, date.GetYears());

}

[TestMethod]

public void CompareDates\_ReturnsDifference()

{

DataTimeLigic date1 = new DataTimeLigic();

DataTimeLigic date2 = new DataTimeLigic();

string result = date1.CompareDates(date2);

Assert.AreEqual("0 год 0 месяцев 0 дней 0 часов 0 минут 0 секунд", result);

}

}

}

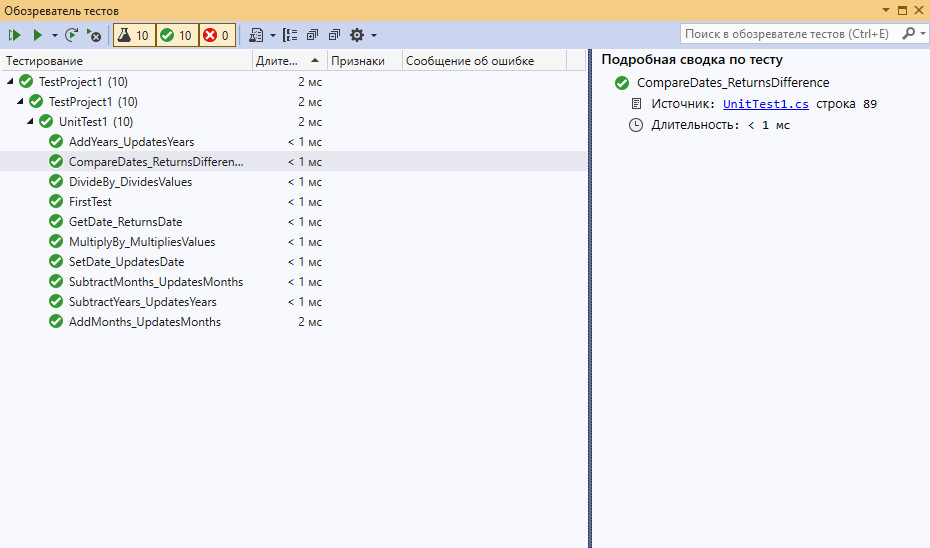


Рисунок 2 - Отображение успешно пройденных тестов

**Вывод:** в ходе выполнения лабораторной работы были изучены и освоены основные принципы построения модульных тестов на основе классов в Visual Studio.