***Министерство образования Республики Беларусь***

***Учреждение образования***

***«Брестский государственный технический университет»***

***Кафедра ИИТ***

**Лабораторная работа №1**

**По дисциплине ППвИС за V семестр**

**Тема: «Повторение основ С#»**

**Выполнил:**

Студент группы ИИ-15 (1)

3-го курса

Волк И. А.

**Проверил:**

Муравьев Г.Л.

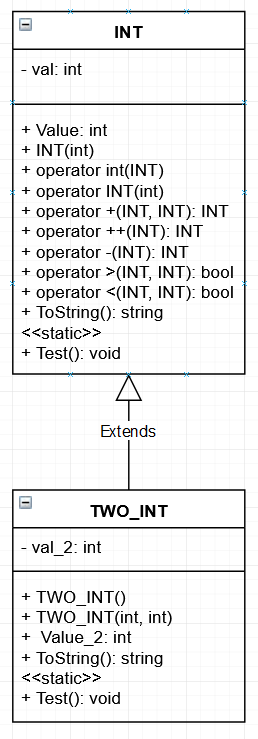
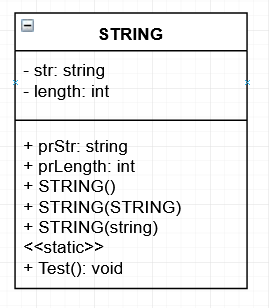
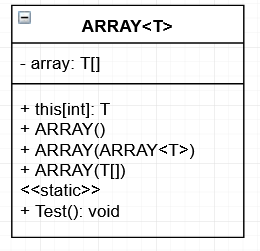
Брест 2019

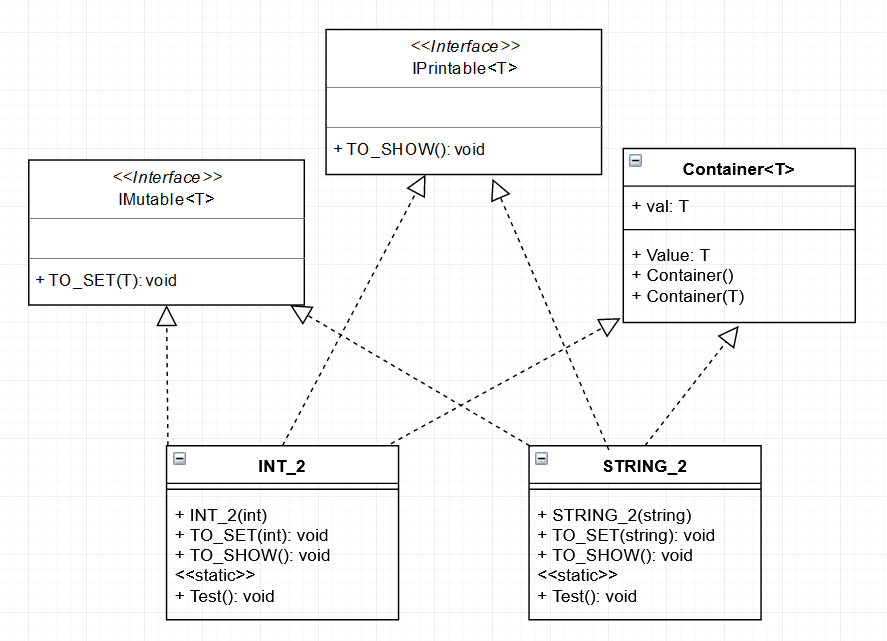
**Цель:** повторить основы языка C#.

**Содержание отчета:**

1. Цель.
2. Содержание отчета.
3. Диаграммы разработанных классов.
4. Результат выполнения программы.
5. Вывод.
6. Листинг программы.

**Диаграммы разработанных классов:**





**Результат выполнения программы:**

**Test INT**

INT: 5

int: 5

I++ + 5: 11

I: 6

-I: -6

5 < 6: True

**Test STRING**

String: message

**Test ARRAY**

5th element of the array: 10

**Test TWO\_INT**

Value #1: 10

Value #2: 20

**Test INT\_2**

Value: 10

**Test STRING\_2**

Value: hello

**Вывод:** в ходе лабораторной работы повторили основы работы с ЯП С#.

**Листинг программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace lab\_1

{

class INT

{

// attributes

private int val;

// properties

public int Value

{

get { return val; }

set { val = value; }

}

// constructors

public INT(int value = 0)

{

val = value;

}

// methods

public static implicit operator int(INT obj)

{

return obj.Value;

}

public static implicit operator INT(int obj)

{

return new INT(obj);

}

public static INT operator +(INT left, INT right)

{

return left.Value + right.Value;

}

public static INT operator ++(INT obj)

{

obj.val++;

return obj;

}

public static INT operator -(INT obj)

{

return -obj.Value;

}

public static bool operator >(INT left, INT right)

{

return left.Value > right.Value;

}

public static bool operator <(INT left, INT right)

{

return left.Value < right.Value;

}

override public string ToString()

{

return val.ToString();

}

public static void Test()

{

Console.WriteLine("\nTest INT\n");

INT i = new INT();

i = 5; // value assignment

int x = i; // to int convertion

Console.WriteLine(string.Format(

"INT: {0}\nint: {1}",

i, // INT

x // int

));

Console.WriteLine(string.Format(

"I++ + 5: {0}",

i++ + 5 // (5++ + 5)

));

Console.WriteLine(string.Format(

"I: {0}\n-I: {1}",

i,

-i

));

Console.WriteLine(string.Format(

"5 < 6: {0}",

new INT(5) < new INT(6)

));

Console.ReadKey();

}

}

class TWO\_INT : INT

{

private int val\_2;

public TWO\_INT() : base() { }

public TWO\_INT(int value\_1, int value\_2)

: base(value\_1)

{

Value\_2 = value\_2;

}

public int Value\_2

{

get

{

return val\_2;

}

set

{

val\_2 = value;

}

}

public override string ToString()

{

return "(" + Value.ToString() + ", " + Value\_2.ToString() + ")";

}

public static new void Test()

{

Console.WriteLine("\nTest TWO\_INT\n");

TWO\_INT ti = new TWO\_INT(10, 20);

Console.WriteLine("Value #1: {0}\nValue #2: {1}",

ti.Value,

ti.Value\_2

);

Console.ReadKey();

}

}

class STRING

{

// attributes

private string str;

private int length;

// properties

public string prStr

{

get

{

return str;

}

set

{

str = value;

length = str.Length;

}

}

public int prLength

{

get

{

return length;

}

private set

{

length = value;

}

}

// constructors

public STRING() : base() { }

public STRING(STRING obj)

{

prStr = obj.prStr;

}

public STRING(string str)

{

prStr = str;

}

// methods

public static void Test()

{

Console.WriteLine("\nTest STRING\n");

STRING str = new STRING("message");

Console.WriteLine("String: {0}", str.prStr);

Console.ReadKey();

}

}

class ARRAY<T>

{

// attributes

private T[] array;

// properties

public T this[int i]

{

get

{

return array[i];

}

set

{

if (i < 0)

throw new IndexOutOfRangeException();

else if (i >= array.Length)

{

T[] arr = new T[i + 1];

Array.Copy(array, arr, array.Length);

arr[i] = value;

array = new T[arr.Length];

Array.Copy(arr, array, arr.Length);

}

array[i] = value;

}

}

// constructors

public ARRAY() : base()

{

array = new T[0];

}

public ARRAY(ARRAY<T> obj)

{

Array.Copy(obj.array, array, obj.array.Length);

}

public ARRAY(T[] arr)

{

array = arr;

}

// methods

public static void Test()

{

Console.WriteLine("\nTest ARRAY\n");

ARRAY<int> arr = new ARRAY<int>();

arr[5] = 10;

Console.WriteLine("5th element of the array: {0}", arr[5]);

Console.ReadKey();

}

}

class Container<T>

{

// attributes

private T val;

// properties

public T Value

{

get { return val; }

set { val = value; }

}

public Container() : base() { }

public Container(T value)

{

val = value;

}

}

interface IMutable<T>

{

void TO\_SET(T value);

}

interface IPrintable

{

void TO\_SHOW();

}

class INT\_2 : Container<int>, IMutable<int>, IPrintable

{

public INT\_2(int value = 2) : base(value) { }

public void TO\_SET(int value)

{

Value = value;

}

public void TO\_SHOW()

{

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

}

public static void Test()

{

Console.WriteLine("\nTest INT\_2\n");

INT\_2 i = new INT\_2();

i.TO\_SET(10);

i.TO\_SHOW();

Console.ReadKey();

}

}

class STRING\_2 : Container<string>, IMutable<string>, IPrintable

{

public STRING\_2(string value = "") : base(value) { }

public void TO\_SET(string value)

{

Value = value;

}

public void TO\_SHOW()

{

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

}

public static void Test()

{

Console.WriteLine("\nTest STRING\_2\n");

STRING\_2 i = new STRING\_2();

i.TO\_SET("hello");

i.TO\_SHOW();

Console.ReadKey();

}

}

class Program

{

static void Main(string[] args)

{

INT.Test();

STRING.Test();

ARRAY<int>.Test();

TWO\_INT.Test();

INT\_2.Test();

STRING\_2.Test();

}

}

}