Учреждение образования "Полоцкий Государственный Университет"

Факультет информационных технологий
Кафедра вычислительных систем и сетей

Отчет по Лабораторной работе № 4

по дисциплине: **"“Объектно-Ориентированные Технологии Программирования и Стандарты Проектирования"**

ВЫПОЛНИЛ

студент группы 18-IT-2
Сыцевич Д.Н.
вариант № 1

ПРОВЕРИЛ

старший преподаватель
Ярошевич П.В.

Полоцк 2020 г.

1 Задача

Создать обобщенный класс, указанной в варианте задания, структуры данных и реализовать дополни- тельные функции для неё. Структура данных должна быть реализованна на основе массива или ссылок

(при реализации нельзя использовать никакие другие коллекции).

2 Вариант № 1

Структура данных "Стек". Дополнительно реализовать функцию проверки структур на равенство.

3 Ход выполнения

Выполнение лабораторной работы включало в себя следующие шаги:

1. Создание класса Stack.
2. Создание класса Program.
3. Компиляция и запуск программы.
4. Написание отчёта.
5. Подготовка к защите.

4 Скриншоты

```
namespace oop1
{
    Ссылок: 14
    class MyStack<T> where T:IComparable
    {

        Ссылок: 16
        private class Data
        {
            private Data next;
            private T value;

            Ссылок: 14
            public Data getNext()
            {
                return next;
            }
            Ссылок: 3
            public void setNext(Data next)
            {
                this.next = next;
            }
            Ссылок: 6
            public T getValue()
            {
                return value;
            }
            Ссылок: 3
            public void setValue(T value)
            {
                this.value = value;
            }

            Ссылок: 2
            public Data()
            {
                value = default;
                next = default;
            }
```

Рис. 1: Скриншот класса Stack

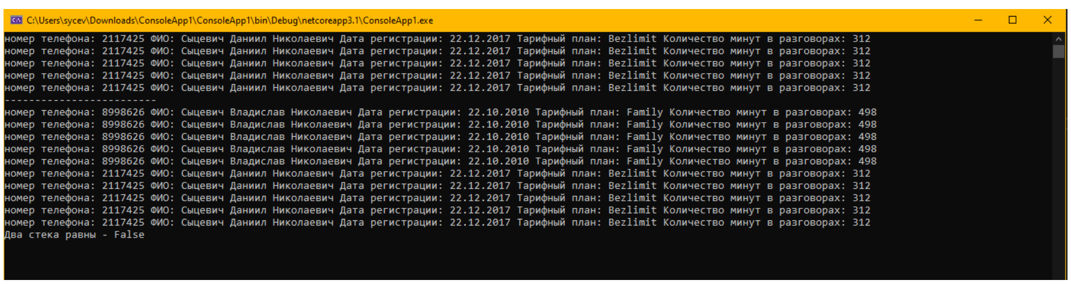

Рис. 2: Скриншот выполнения с примитивными типами данных

5 Source Code

MyStack.cs

```
  using    System;
using System.Collections.Generic;
using System.Text;

namespace oop1
{
        class MyStack<T> where T:IComparable
        {

                private class Data
                {
                        private Data next;
                        private T value;

                        public Data getNext()
                        {
                                return next;
                        }
                        public void setNext(Data next)
                        {
                                this.next = next;
                        }
                        public T getValue()
                        {
                                return value;
                        }
                        public void setValue(T value)
                        {
                                this.value = value;
                        }

                        public Data()
                        {
                                value = default;
                                next = default;
                        }
```

```
        public String toString()
        {
                return value + "␣";
        }
};

private Data head;
private Data start;

public MyStack()
{
        start = new Data();
        head = start;
}

public void push(T obj)
{
        Data ne = new Data();
        ne.setValue(obj);
        head.setNext(ne);
        head = ne;
}

public T pop()
{
        if (head == start) return default;
        Data oh = head;
        Data iterator = start;
        while (iterator.getNext() != null && iterato
        {
                iterator = iterator.getNext();
        }
        head = iterator;
        head.setNext(null);
        return oh.getValue();

}
public void clear()
{
        while (head != start)
```

```
        {
                Data iterator = start;
                while (iterator.getNext() != null &&
                {
                        iterator = iterator.getNext(
                }
                head = iterator;
                head.setNext(null);
        }
}
public void print()
{
        if (head == start) return;
        Data iterator = start.getNext();
        while (iterator != null)
        {
                Console.WriteLine(iterator.toString(
                iterator = iterator.getNext();
        }
}

public MyStack<T> clone()
{
        if (head == start) return new MyStack<T>();
        MyStack<T> cl = new MyStack<T>();
        for (Data iterator = start.getNext(); iterat
        {
                cl.push(iterator.getValue());
        }
        return cl;
}

public bool isEmpty()
{
        return head == start;
}

public override bool Equals(object obj)
{
        if (!(obj is MyStack<T>)) return false;
```

```
                MyStack<T> sobj = (MyStack <T>)obj;
                MyStack<T> mcl = clone();
                while(sobj.head!=sobj.start||mcl.head!=mcl.s
                {
                        if (!sobj.pop().Equals(mcl.pop()))
                                return false;
                }
                if (sobj.head == sobj.start || mcl.head == n
                        return true;
                return false;
        }

        public void sort()
        {
                if (head == start) return;
                for (Data iterator = start.getNext(); iterat
                {
                        for (Data insit = start.getNext(); i
                        {
                                if (iterator.getValue().Comp
                                {
                                        T temp = iterator.ge
                                        iterator.setValue(in
                                        insit.setValue(temp)
                                }
                        }
                }
        }
    }

}
```

Program.cs

```
  using    System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace oop1
```

```
{
    class phone : IComparable
    {
        protected internal int nomer; //
        protected internal string FIO; //stroka
        protected internal string date; //data (dd.mm.yyyy)
        protected internal string tarif; //stroka
        protected internal int minut; //chislo
        protected internal phone(int nomer, string FIO, string date,
        {
            this.nomer = nomer;
            this.FIO = FIO;
            this.date = date;
            this.tarif = tarif;
            this.minut = minut;
        }

        protected internal phone()
        {
            nomer = 2117425;
            FIO = "                    ␣                    ␣                    "
            date = "22.12.2017";
            tarif = "Bezlimit";
            minut = 312;
        }

        protected internal phone(phone copy)
        {
            nomer = copy.nomer;
            FIO = copy.FIO;
            date = copy.date;
            tarif = copy.tarif;
            minut = copy.minut;
        }

        protected void SetNomer(int nomer)
        { this.nomer = nomer; }

        protected int GetNomer()
        { return nomer; }
```

```
protected void SetFIO(string FIO)
{ this.FIO = FIO; }

protected string GetFIO()
{ return FIO; }

protected void SetDate(string date)
{ this.date = date; }

protected string GetDate()
{ return date; }

protected void SetTarif(string tarif)
{ this.tarif = tarif; }

protected string GetTarif()
{ return tarif; }

protected void SetMinut(int minut)
{ this.minut = minut; }

protected int GetMinut()
{ return minut; }


 public override bool Equals(object obj)
 {
     if (obj == null)
     return false;
     phone m = obj as phone; //                              null
     if (m as phone == null)
     return false;
    return m.nomer == this.nomer && m.FIO == this.FIO && m.d
 }

public override string ToString()
{
    return "                ␣                    :␣" + nomer
    + "␣       :␣" + FIO
```

```
            + "␣          ␣                          :␣" + date
            + "␣                    ␣          :␣" + tarif
            + "␣                        ␣          ␣  ␣
        }

        public int CompareTo(object obj)
        {
            if (nomer > ((phone)obj).nomer)
                return 1;
            else if (nomer == ((phone)obj).nomer)
                return 0;
            else
                return -1;
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            phone phone1 = new phone();
            phone phone2 = new phone(8998626, "                    ␣
            MyStack<phone> stack1 = new MyStack<phone>();
            MyStack<phone> stack2 = new MyStack<phone>();
            for(int i = 0;i<5;i++)
            {
                stack1.push(phone2);
                stack2.push(phone2);
            }
            MyStack<phone> cl = stack1.clone();
            while(!cl.isEmpty())
            {
                stack2.push(cl.pop());
            }
            stack1.print();
            Console.WriteLine("-------------------------------");
            stack2.print();
            Console.WriteLine("   " +
                "    ␣              ␣          ␣-␣{0}\n\n",stack1.Equals
            Console.ReadKey();
```

```
            }
        }
    }
```