Учреждение образования "Полоцкий Государственный Университет"

Факультет информационных технологий
Кафедра вычислительных систем и сетей

Отчет по Лабораторной работе № 2

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

ВЫПОЛНИЛ

студент группы 18-ИТ-1
Сапего Н.Ю.
вариант № 11

ПРОВЕРИЛ

старший преподаватель
Ярошевич П.В.

Полоцк 2020 г.

1 Задача

Для сущностей, согласно варианту задания, необходимо определить и реализовать иерархию классов. Для этого среди сущностей необходимо выбрать ту, наследниками которой будут является все остальные.
К каждому классу предявляются требования к реализации класса, аналогично требованиям к реализации класса из предыдущей лабораторной работы.
Базовый класс содержит общие для всех классов наследников поля (они должны быть доступны лишь наследникам, 2-4 атрибута) и является абстрактным. Каждый класс наследник по своему расширяет свойства базового класса (3-6 атрибута должно быть в дочернем классе, с учетом атрибутов базового класса). Базовый класс должен содержать статическую переменную отвечающую за количество созданных объектов.

2 Вариант № 11

Республика, монархия, королевство, государство.

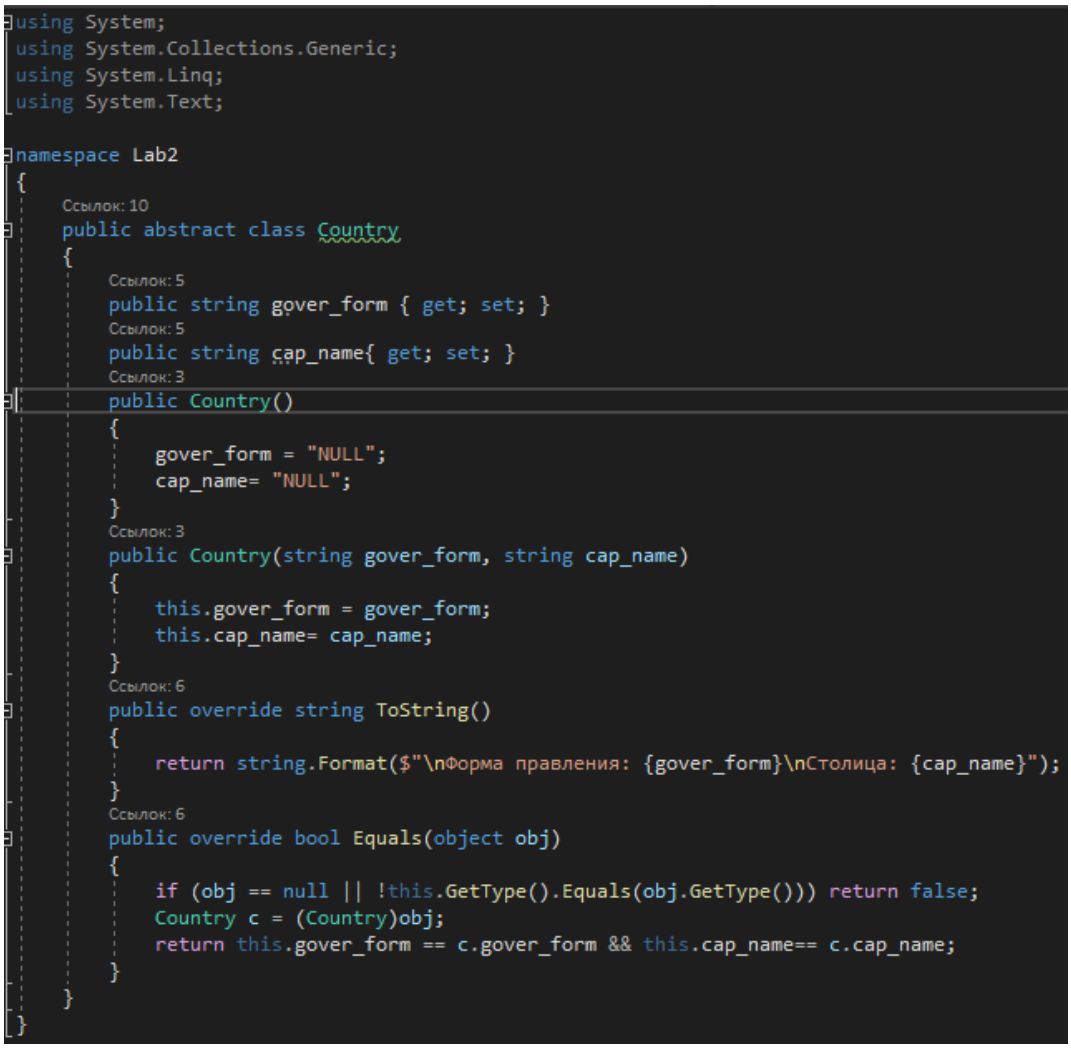
3 Ход выполнения

Написать демонстрационную программу, в которой создаются объекты реализованных классов и помещаются в массив (размер массива не превышает 10). Допускается создание объектов на основе введенных данных. Затем осуществляется вывод элементов массива в консоль.

4 Скриншоты

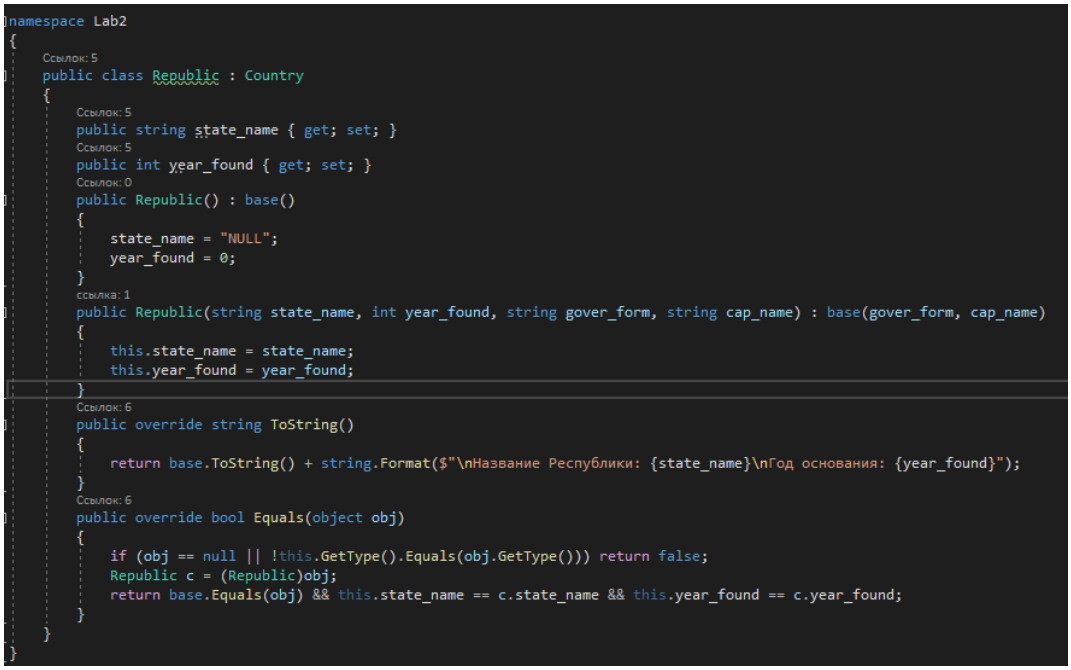
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab2
{
    Ссылок: 0
    class Program
    {
        Ссылок: 0
        public static void Main(string [] args)
        {
            Country [] a = new Country[3];
            a[0] = new Republic("Республика Беларусь", 1919, "Республика", "Минск");
            a[1] = new Kingdom("Великобритания","Королева", "Королевство","Лондон");
            a[2] = new Monarchy("Австралия", "Королева", "Конституционная монархия", "Канберра");

            Console.Write("Вывод всех записей:\n");
            foreach (Country o in a)
            {
                Console.WriteLine(o);
            }
        }
    }
}
```

Рис. 1: Главный класс программы



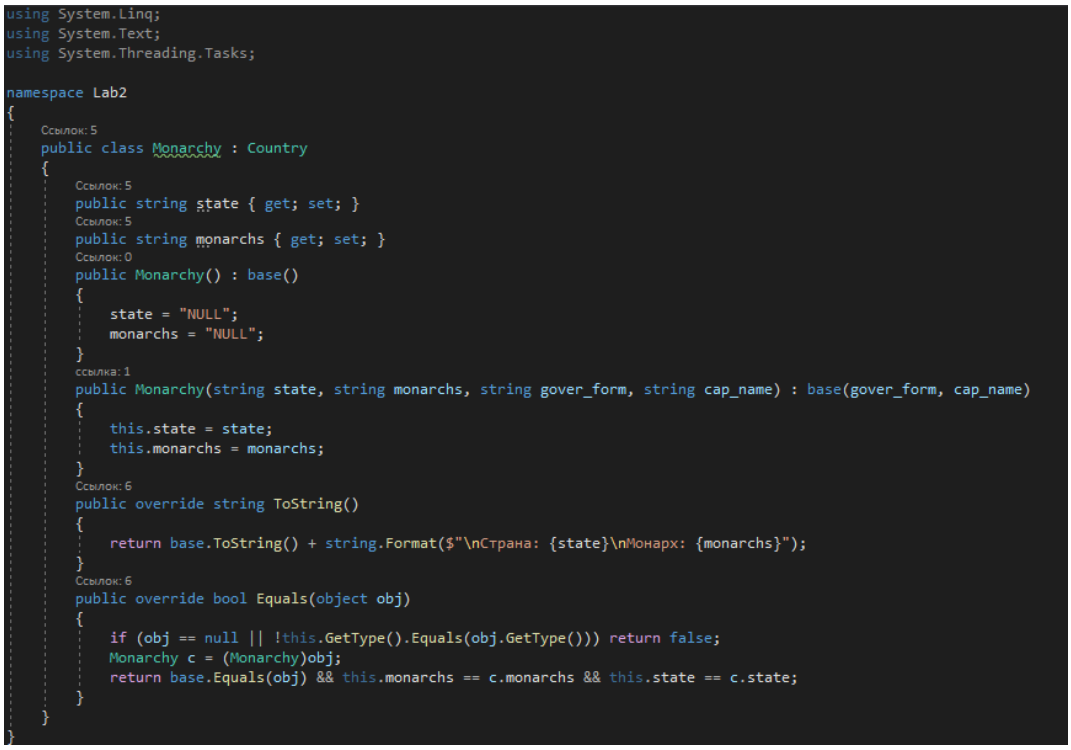
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Lab2
{
    Ссылок: 10
    public abstract class Country
    {
        Ссылок: 5
        public string gover_form { get; set; }
        Ссылок: 5
        public string cap_name{ get; set; }
        Ссылок: 3
        public Country()
        {
            gover_form = "NULL";
            cap_name= "NULL";
        }
        Ссылок: 3
        public Country(string gover_form, string cap_name)
        {
            this.gover_form = gover_form;
            this.cap_name= cap_name;
        }
        Ссылок: 6
        public override string ToString()
        {
            return string.Format($"\nФорма правления: {gover_form}\nСтолица: {cap_name}");
        }
        Ссылок: 6
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())) return false;
            Country c = (Country)obj;
            return this.gover_form == c.gover_form && this.cap_name== c.cap_name;
        }
    }
}
```

Рис. 2: Производный класс Country



```
namespace Lab2
{
    Ссылок: 5
    public class Republic : Country
    {
        Ссылок: 5
        public string state_name { get; set; }
        Ссылок: 5
        public int year_found { get; set; }
        Ссылок: 0
        public Republic() : base()
        {
            state_name = "NULL";
            year_found = 0;
        }
        ссылка: 1
        public Republic(string state_name, int year_found, string gover_form, string cap_name) : base(gover_form, cap_name)
        {
            this.state_name = state_name;
            this.year_found = year_found;
        }
        Ссылок: 6
        public override string ToString()
        {
            return base.ToString() + string.Format($"\nНазвание Республики: {state_name}\nГод основания: {year_found}");
        }
        Ссылок: 6
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())) return false;
            Republic c = (Republic)obj;
            return base.Equals(obj) && this.state_name == c.state_name && this.year_found == c.year_found;
        }
    }
}
```

Рис. 3: Производный класс Republic



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

namespace Lab2
{
    Ссылок: 5
    public class Monarchy : Country
    {
        Ссылок: 5
        public string state { get; set; }
        Ссылок: 5
        public string monarchs { get; set; }
        Ссылок: 0
        public Monarchy() : base()
        {
            state = "NULL";
            monarchs = "NULL";
        }
        ссылка: 1
        public Monarchy(string state, string monarchs, string gover_form, string cap_name) : base(gover_form, cap_name)
        {
            this.state = state;
            this.monarchs = monarchs;
        }
        Ссылок: 6
        public override string ToString()
        {
            return base.ToString() + string.Format($"\nСтрана: {state}\nМонарх: {monarchs}");
        }
        Ссылок: 6
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())) return false;
            Monarchy c = (Monarchy)obj;
            return base.Equals(obj) && this.monarchs == c.monarchs && this.state == c.state;
        }
    }
}
```

Рис. 4: Производный класс Monarchy

```
namespace Lab2
{
    Ссылок: 5
    public class Kingdom : Country
    {
        Ссылок: 5
        public string state_name { get; set; }
        Ссылок: 5
        public string state_head { get; set; }
        Ссылок: 0
        public Kingdom() : base()
        {
            state_name = "NULL";
            state_head = "NULL";
        }
        ссылка: 1
        public Kingdom(string state_name, string capital, string gover_form, string cap_name) : base(gover_form, cap_name)
        {
            this.state_name = state_name;
            this.state_head = capital;
        }
        Ссылок: 6
        public override string ToString()
        {
            return base.ToString() + string.Format($"\nНазвание Государства: {state_name}\nГлава: {state_head}\n");
        }
        Ссылок: 6
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())) return false;
            Kingdom c = (Kingdom)obj;
            return base.Equals(obj) && this.state_name == c.state_name && this.state_head == c.state_head;
        }
    }
}
```

Рис. 5: Производный класс Kingdom

```
C:\Windows\system32\cmd.exe
Вывод всех записей:

Форма правления: Республика
Столица: Минск
Название Республики: Республика Беларусь
Год основания: 1919

Форма правления: Королевство
Столица: Лондон
Название Государства: Великобритания
Глава: Королева


Форма правления: Конституционная монархия
Столица: Канберра
Страна: Австралия
Монарх: Королева
Для продолжения нажмите любую клавишу . . .
```

Рис. 6: Результат выполнения программы

5 Source Code

Source.cs

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

namespace Lab2
{
    class Program
    {
        public static void Main(string [] args)
        {
            Country [] a = new Country[3];
            a[0] = new Republic("
            a[1] = new Kingdom("                                    ","
            a[2] = new Monarchy("                        ", "

            Console.Write("                                         :\n");
            foreach (Country o in a)
            {
                Console.WriteLine(o);
            }
        }
    }
}
```

Country.cs

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

namespace Lab2
{
    public abstract class Country
    {
        public string gover_form { get; set; }
```

```
        public string cap_name{ get; set; }
        public Country()
        {
            gover_form = "NULL";
            cap_name= "NULL";
        }
        public Country(string gover_form, string cap_name)
        {
            this.gover_form = gover_form;
            this.cap_name= cap_name;
        }
        public override string ToString()
        {
            return string.Format($"\n                                                    :
        }
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())
            Country c = (Country)obj;
            return this.gover_form == c.gover_form && this.cap_name=
        }
    }
}
```

Republic.cs

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

namespace Lab2
{
    public class Republic : Country
    {
        public string state_name { get; set; }
        public int year_found { get; set; }
        public Republic() : base()
        {
            state_name = "NULL";
```

```
                year_found = 0;
        }
        public Republic(string state_name, int year_found, string go
        {
            this.state_name = state_name;
            this.year_found = year_found;
        }
        public override string ToString()
        {
            return base.ToString() + string.Format($"\n
        }
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())
            Republic c = (Republic)obj;
            return base.Equals(obj) && this.state_name == c.state_na
        }
    }
}
```

Monarchy.cs

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

namespace Lab2
{
    public class Monarchy : Country
    {
            public string state { get; set; }
        public string monarchs { get; set; }
        public Monarchy() : base()
        {
                state = "NULL";
                monarchs = "NULL";
        }
        public Monarchy(string state, string monarchs, string gover_
        {
```

```
                this.state = state;
            this.monarchs = monarchs;
        }
        public override string ToString()
        {
            return base.ToString() + string.Format($"\n            :
        }
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())
            Monarchy c = (Monarchy)obj;
            return base.Equals(obj) && this.monarchs == c.monarchs &
        }
    }
}
```

Kingdom.cs

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

namespace Lab2
{
    public class Kingdom : Country
    {
        public string state_name { get; set; }
        public string state_head { get; set; }
        public Kingdom() : base()
        {
            state_name = "NULL";
            state_head = "NULL";
        }
        public Kingdom(string state_name, string capital, string gov
        {
            this.state_name = state_name;
                this.state_head = capital;
        }
        public override string ToString()
```

```
        {
            return base.ToString() + string.Format($"\n
        }
        public override bool Equals(object obj)
        {
            if (obj == null || !this.GetType().Equals(obj.GetType())
            Kingdom c = (Kingdom)obj;
                        return base.Equals(obj) && this.state_name =
        }
    }
}
```