

## Оглавление

Оглавление .....	2
1 Задание .....	3
2 Код программы .....	3
3 Результат работы консольного приложения.....	8
4 Ответы на контрольные вопросы.....	9
5 Выводы .....	11

## 1 Задание

Создать консольное приложение на языке C# с заданным классом, а также массив объектов этого класса. Выполнить заданные запросы LINQ к массиву объектов.

Билет на междугородный автобус. Код, рейс, пункт назначения, время отправления, длительность, номер места, код автобуса. Автобус. Код, модель, число посадочных мест.

1. Данные по всем билетам с пунктом назначения «.».
2. Рейсы с временем отправления «...».
3. Число билетов с номерами мест от «.» до «.».
4. Пункты назначения и рейсы с длительностью более «.».
5. Средняя длительность рейсов в пункт назначения «.».
6. Все билеты, сгруппированные по пункту назначения (group).
7. Рейсы и места с указанием модели автобуса и числа мест в нем (join).

## 2 Код программы

```
using System;
using System.Linq;

namespace ЛР_1
{
    // Описание класса Ticket (билет)
    class Ticket
    {
        // Автоматические свойства класса Ticket
        public string TicketID { get; set; } // Код билета
        public int Trip { get; set; } // Рейс
        public string Point { get; set; } // Пункт назначения
        public string Departure { get; set; } // Время отправления
        public int Duration { get; set; } // Длительность
        public int Number { get; set; } // Номер места
        public string BusID { get; set; } // Код автобуса

        // Переопределение метода ToString для вывода информации об объектах
        public override string ToString()
        {
            return string.Format("- Код билета: {0}\n" +
                "Рейс: {1}\n" +
                "Пункт назначения: {2}; " +
                "Время отправления: {3}; " +
                "Длительность: {4} мин; " +
                "Номер места: {5}; " +
                "Код автобуса: {6}.",
                TicketID,
                Trip,
                Point,
                Departure,
                Duration,
                Number,
                BusID);
        }
    }
}
```

```

        Point,
        Departure,
        Duration,
        Number,
        BusID
    );
}
}
// Описание класса Bus (автобус)
class Bus
{
    // Автоматические свойства класса Bus
    public string BusID { get; set; } // Код автобуса
    public string Model { get; set; } // Модель
    public int NumberOfSeats { get; set; } // Число посадочных мест
}
internal class Program
{
    static void Main(string[] args)
    {
        Console.Title = "Выполнение запросов LINQ к массиву объектов";
        // Объявление массива объектов класса Ticket с инициализацией элементов
        Ticket[] ticketsOnTrip = new[]
        {
            new Ticket { TicketID = "ИБ116",
                        Trip = 307,
                        Point = "Иваново",
                        Departure = "08:30",
                        Duration = 120,
                        Number = 14,
                        BusID = "A071"
            },
            new Ticket { TicketID = "КС319",
                        Trip = 418,
                        Point = "Кострома",
                        Departure = "08:30",
                        Duration = 100,
                        Number = 05,
                        BusID = "A019"
            },
            new Ticket { TicketID = "ИБ414",
                        Trip = 312,
                        Point = "Иваново",
                        Departure = "13:40",
                        Duration = 130,
                        Number = 20,
                        BusID = "B027"
            },
            new Ticket { TicketID = "РС514",
                        Trip = 207,
                        Point = "Ростов",
                        Departure = "16:00",
                        Duration = 80,
                        Number = 31,
                        BusID = "A071"
            },
            new Ticket { TicketID = "ВЛ907",
                        Trip = 425,
                        Point = "Владимир",
                        Departure = "18:40",
                        Duration = 200,
                        Number = 19,
                        BusID = "A019"
            },
            new Ticket { TicketID = "КС403",

```

```

        Trip = 315,
        Point = "Кострома",
        Departure = "20:20",
        Duration = 80,
        Number = 27,
        BusID = "A071"
    },
};
Console.WriteLine("Результаты запросов LINQ");
// Вызовы методов выполнения запросов LINQ
GetIv(ticketsOnTrip);
GetTripTime(ticketsOnTrip);
GetNumTicket(ticketsOnTrip);
GetPointTrip(ticketsOnTrip);
GetAverDur(ticketsOnTrip);
GetTicPoint(ticketsOnTrip);
GetTripNum(ticketsOnTrip);
Console.ReadLine();
}
// Метод получения данных по всем билетам с пунктом назначения Иваново
static void GetIv(Ticket[] tickets)
{
    Console.WriteLine("\n1. Данные по всем билетам с пунктом назначения Иваново: ");
    var iv = from t in tickets
              where t.Point == "Иваново"
              select t;
    foreach (var i in iv)
    {
        Console.WriteLine(i.ToString());
    }
}
// Метод получения номеров рейсов с временем отправления 08:30
static void GetTripTime(Ticket[] tickets)
{
    Console.WriteLine("\n2. Рейсы с временем отправления 08:30: ");
    var trip = from t in tickets
                where t.Departure == "08:30"
                select t.Trip;
    foreach (var t in trip)
    {
        Console.WriteLine("- {0}", t);
    }
}
// Метод получения количества билетов с номерами мест от 5 до 25
static void GetNumTicket(Ticket[] tickets)
{
    Console.WriteLine("\n3. Количество билетов с номерами мест от 5 до 25:");
    var num = from t in tickets
                where t.Number >= 5 && t.Number <= 25
                select t.Number;
    int numTickets = num.Count();
    Console.WriteLine("Всего {0} билета", numTickets);
}
// Метод получения пунктов назначения и рейсов с длительностью более 100 мин
static void GetPointTrip(Ticket[] tickets)
{
    Console.WriteLine("\n4. Пункты назначения и рейсы с длительностью более 100
мин:");
    var pointTrip = from t in tickets
                     where t.Duration > 100
                     select t;
    foreach (var p in pointTrip)
    {
        Console.WriteLine("- {0}, рейс {1}", p.Point, p.Trip);
    }
}

```

```

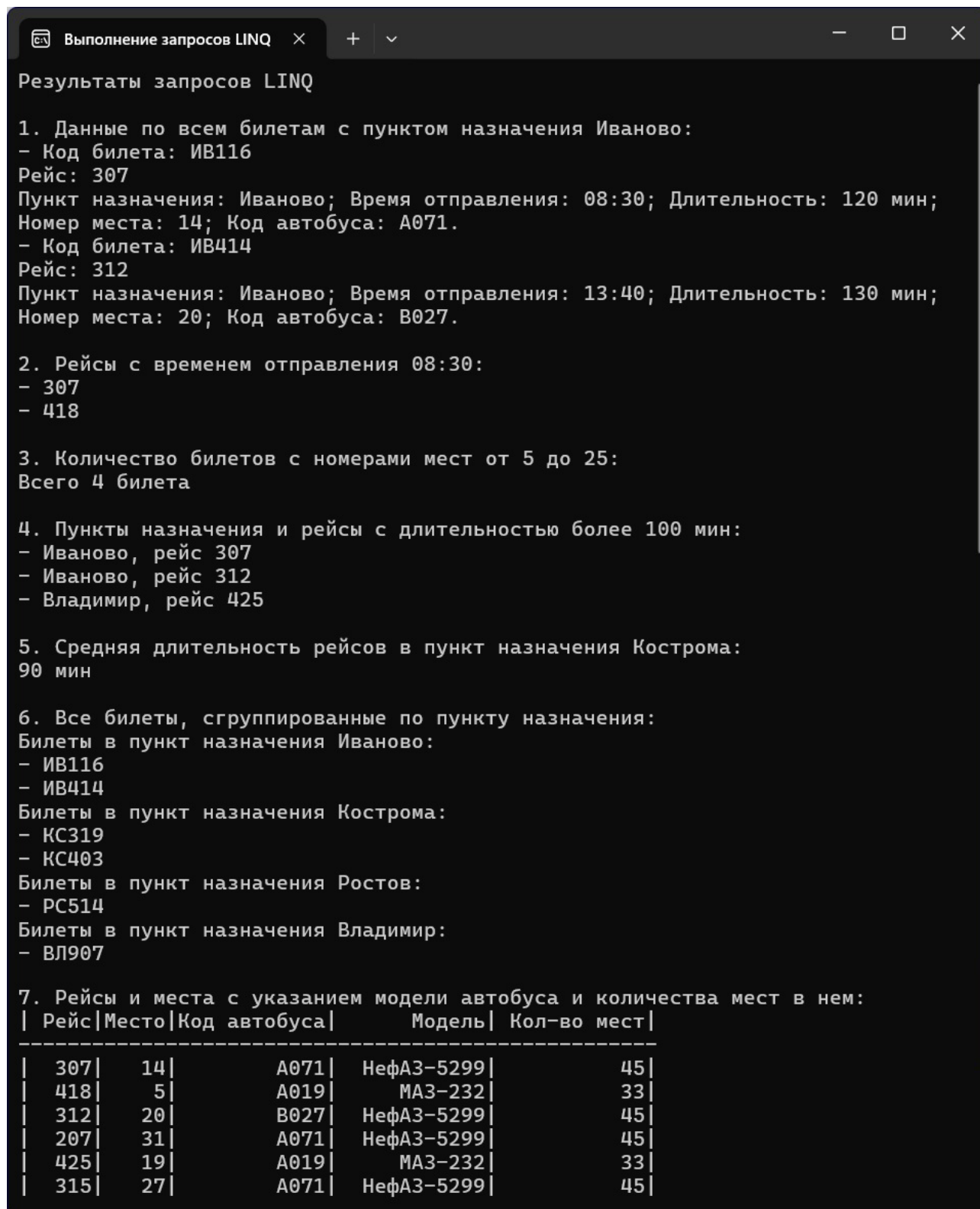
}
// Метод нахождения средней длительности рейсов в пункт назначения Кострома
static void GetAverDur(Ticket[] tickets)
{
    Console.WriteLine("\n5. Средняя длительность рейсов в пункт назначения
                        Кострома:");
    var averDur = from t in tickets
                  where t.Point == "Кострома"
                  select t.Duration;
    Console.WriteLine("{0} мин", averDur.Average());
}
// Метод нахождения билетов, сгруппированных по пункту назначения
static void GetTicPoint(Ticket[] tickets)
{
    Console.WriteLine("\n6. Все билеты, сгруппированные по пункту назначения:");
    var groups = from t in tickets
                 group t by t.Point;
    // Цикл для выбора каждой группы group из списка групп groups
    foreach (var group in groups)
    {
        Console.WriteLine("Билеты в пункт назначения {0}:", group.Key);
        // Цикл для выбора каждого элемента elem группы group
        foreach (var elem in group)
        {
            Console.WriteLine("- " + elem.TicketID);
        }
    }
}
// Метод нахождения рейсов и мест с указанием модели автобуса и числа мест в нем
static void GetTripNum(Ticket[] tickets)
{
    // Массив объектов класса Bus
    Bus[] buses = new[]
    {
        new Bus
        {
            BusID = "A071",
            Model = "НефАЗ-5299",
            NumberOfSeats = 45
        },
        new Bus
        {
            BusID = "A019",
            Model = "МАЗ-232",
            NumberOfSeats = 33
        },
        new Bus
        {
            BusID = "B027",
            Model = "НефАЗ-5299",
            NumberOfSeats = 45
        }
    };
    Console.WriteLine("\n7. Рейсы и места с указанием модели автобуса и количества
                        мест в нем:");
    var tripNum = from t in tickets
                  join b in buses on t.BusID equals b.BusID
                  select new
                  {
                      trip = t.Trip,
                      numb = t.Number,
                      bus = t.BusID,
                      mod = b.Model,
                      nums = b.NumberOfSeats
                  };
};

```

```
Console.WriteLine("{0,5}|{1,5}|{2,12}|{3,12}|{4,12}|",  
    "Рейс", "Место", "Код автобуса", "Модель", "Кол-во мест");  
Console.WriteLine(new String('-', 52));  
foreach (var tn in tripNum)  
{  
    Console.WriteLine("{0,5}|{1,5}|{2,12}|{3,12}|{4,12}|",  
        tn.trip, tn.numb, tn.bus, tn.mod, tn.nums);  
}  
}  
}
```

### 3 Результат работы консольного приложения

Результат работы консольного приложения представлен на рисунке 1.



```
Выполнение запросов LINQ × + ▾
Результаты запросов LINQ

1. Данные по всем билетам с пунктом назначения Иваново:
- Код билета: IB116
Рейс: 307
Пункт назначения: Иваново; Время отправления: 08:30; Длительность: 120 мин;
Номер места: 14; Код автобуса: A071.
- Код билета: IB414
Рейс: 312
Пункт назначения: Иваново; Время отправления: 13:40; Длительность: 130 мин;
Номер места: 20; Код автобуса: B027.

2. Рейсы с временем отправления 08:30:
- 307
- 418

3. Количество билетов с номерами мест от 5 до 25:
Всего 4 билета

4. Пункты назначения и рейсы с длительностью более 100 мин:
- Иваново, рейс 307
- Иваново, рейс 312
- Владимир, рейс 425

5. Средняя длительность рейсов в пункт назначения Кострома:
90 мин

6. Все билеты, сгруппированные по пункту назначения:
Билеты в пункт назначения Иваново:
- IB116
- IB414
Билеты в пункт назначения Кострома:
- KC319
- KC403
Билеты в пункт назначения Ростов:
- RC514
Билеты в пункт назначения Владимир:
- VL907

7. Рейсы и места с указанием модели автобуса и количества мест в нем:
| Рейс|Место|Код автобуса|      Модель| Кол-во мест|
-----|-----|-----|-----|-----|
| 307| 14|      A071| НефАЗ-5299|      45|
| 418|  5|      A019|   МАЗ-232|      33|
| 312| 20|      B027| НефАЗ-5299|      45|
| 207| 31|      A071| НефАЗ-5299|      45|
| 425| 19|      A019|   МАЗ-232|      33|
| 315| 27|      A071| НефАЗ-5299|      45|
```

Рисунок 1 – Результат работы консольного приложения

## Оглавление

Оглавление .....	2
1 Создание XML-документа с помощью консольного приложения на языке C#. 3	
1.1 Задание .....	3
1.2 Код программы .....	3
1.3 Результат работы консольного приложения .....	4
2 Создание XML-документа из массива объектов с помощью консольного приложения на языке C# .....	4
2.1 Задание .....	4
2.2 Код программы .....	4
2.3 Результат работы консольного приложения .....	5
3 Выполнение запросов LINQ к документу XML с помощью консольного приложения на языке C# .....	6
3.1 Задание .....	6
3.2 Код программы .....	6
3.3 Результат работы консольного приложения .....	10
4 Ответы на контрольные вопросы.....	11
5 Выводы .....	13



# 1 Создание XML-документа с помощью консольного приложения на языке C#

## 1.1 Задание

Требуется разработать консольное приложение на языке C#, в котором с помощью классов System.Xml.Linq «с нуля» создается одна ветвь XML-документа.

## 1.2 Код программы

Разработаем приложение, содержащее следующий код:

```
using System.Xml.Linq;

namespace ЛР_2_1
{
    class Program
    {
        static void Main(string[] args)
        {
            XDocument xmlDoc =
                new XDocument(
                    new XDeclaration("1.0", "utf-8", "yes"),
                    new XComment("Данные о предоставляемых банковских услугах"),
                    new XElement("банковские_услуги",
                        new XElement("банк",
                            new XAttribute("бик", "044525225"),
                            new XAttribute("название", "Сбер"),
                            new XElement("вклад",
                                new XAttribute("код", "сб137"),
                                new XElement("вид", "срочный"),
                                new XElement("процент", "11,6",
                                    new XAttribute("ед_изм", "%")),
                                new XElement("вкладчик",
                                    new XAttribute("инн", "374981357822"),
                                    new XElement("фио", "Воробьев Сергей Борисович"),
                                    new XElement("сумма", "150000,00",
                                        new XAttribute("ед_изм", "руб."))
                                )
                            )
                        )
                    )
                );
            xmlDoc.Save("bank.xml");
        }
    }
}
```

### 1.3 Результат работы консольного приложения

После выполнения метода Main() консольного приложения в файл «bank.xml» записаны следующие данные:

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!--Данные о предоставляемых банковских услугах-->
<банковские_услуги>
  <банк бик="044525225" название="Сбер">
    <вклад код="сб137">
      <вид>срочный</вид>
      <процент ед_изм="%">11,6</процент>
      <вкладчик инн="374981357822">
        <фio>Воробьев Сергей Борисович</фio>
        <сумма ед_изм="руб.">150000,00</сумма>
      </вкладчик>
    </вклад>
  </банк>
</банковские_услуги>
```

## 2 Создание XML-документа из массива объектов с помощью консольного приложения на языке C#

## 2.1 Задание

Требуется добавить метод в приложение из лабораторной работы №1 (запросы к массиву объектов), который с помощью запроса LINQ преобразовывает данные из массива объектов в XML.

## 2.2 Код программы

Добавим в разработанное в предыдущей лабораторной работе приложение метод для создания XML-документа из массива объектов:

[illegible]

```

        new XElement("point", t.Point),
        new XElement("departure", t.Departure),
        new XElement("duration", t.Duration, new XAttribute("unit",
            "мин")),
        new XElement("number", t.Number)
    );
    // Корневой элемент
    XElement rootElem = new XElement("ticketBus", xmlData);
    // Документ XML
    XDocument xmlDoc = new XDocument();
    xmlDoc.Add(rootElem);
    xmlDoc.Save("ticket.xml");
}

```

## 2.3 Результат работы консольного приложения

В результате работы приложения получен XML-документ «ticket.xml», содержащий следующий код:

```

<?xml version="1.0" encoding="utf-8"?>
<ticketBus>
  <ticket id="ИБ116" trip="307">
    <point>Иваново</point>
    <departure>08:30</departure>
    <duration unit="мин">120</duration>
    <number>14</number>
  </ticket>
  <ticket id="КС319" trip="418">
    <point>Кострома</point>
    <departure>08:30</departure>
    <duration unit="мин">100</duration>
    <number>5</number>
  </ticket>
  <ticket id="ИБ414" trip="312">
    <point>Иваново</point>
    <departure>13:40</departure>
    <duration unit="мин">130</duration>
    <number>20</number>
  </ticket>
  <ticket id="РС514" trip="207">
    <point>Ростов</point>
    <departure>16:00</departure>
    <duration unit="мин">80</duration>
    <number>31</number>
  </ticket>
  <ticket id="ВЛ907" trip="425">
    <point>Владимир</point>
    <departure>18:40</departure>
    <duration unit="мин">200</duration>
    <number>19</number>
  </ticket>
  <ticket id="КС403" trip="315">
    <point>Кострома</point>
    <departure>20:20</departure>
    <duration unit="мин">80</duration>
    <number>27</number>
  </ticket>
</ticketBus>

```

## 3 Выполнение запросов LINQ к документу XML с помощью консольного приложения на языке C#

### 3.1 Задание

Разработать консольное приложение на языке C#, которое обеспечивает выполнение заданных запросов к XML-документу. Запрос №3 требуется сделать с использованием операции group, запрос №4 – с помощью join, а запрос №5 – с помощью выражения на языке XPath.

Данные для разработки запросов LINQ к XML-документу.

Банк

1. Вид и сумма вкладов с процентной ставкой более «Процент».
2. Число вкладов, по которым процент составляет от «.» до «.».
3. Данные по вкладам размером более «.», сгруппированные по клиентам.
4. Список вкладов с указанием данных по вкладчику (join).
5. Клиенты, сделавшие более одного вклада (XPath).

### 3.2 Код программы

Составим XML-документ следующего содержания:

```
<?xml version="1.0" encoding="utf-8"?>
<банковские_услуги>
  <банк бик="044525225" название="Сбербанк">
    <вклад код="с6137">
      <вид>срочный</вид>
      <процент ед_изм="%">11,6</процент>
      <вкладчик инн="374981357822">
        <фio>Воробьев Сергей Борисович</фio>
        <сумма ед_изм="руб.">150000,00</сумма>
      </вкладчик>
      <вкладчик инн="761594352278">
        <фio>Прокопьев Антон Сергеевич</фio>
        <сумма ед_изм="руб.">275000,00</сумма>
      </вкладчик>
      <вкладчик инн="762248659912">
        <фio>Бутенко Алексей Викторович</фio>
        <сумма ед_изм="руб.">680000,00</сумма>
      </вкладчик>
    </вклад>
    <вклад код="с6265">
      <вид>накопительный</вид>
      <процент ед_изм="%">15,3</процент>
      <вкладчик инн="117856842341">
        <фio>Копылов Семен Андреевич</фio>
        <сумма ед_изм="руб.">1250000,00</сумма>
      </вкладчик>
    </вклад>
  </банк>
</банковские_услуги>
```

```

<вклад код="с6318">
  <вид>социальный</вид>
  <процент ед_изм="%">1,95</процент>
  <вкладчик инн="374981357822">
    <фio>Воробьев Сергей Борисович</фio>
    <сумма ед_изм="руб.">150000,00</сумма>
  </вкладчик>
  <вкладчик инн="762248659912">
    <фio>Бутенко Алексей Викторович</фio>
    <сумма ед_изм="руб.">680000,00</сумма>
  </вкладчик>
</вклад>
</банк>
<банк бик="044525974" название="Тинькофф">
  <вклад код="тн175">
    <вид>до востребования</вид>
    <процент ед_изм="%">5,16</процент>
    <вкладчик инн="374981357822">
      <фio>Воробьев Сергей Борисович</фio>
      <сумма ед_изм="руб.">150000,00</сумма>
    </вкладчик>
    <вкладчик инн="761594352278">
      <фio>Прокопьев Антон Сергеевич</фio>
      <сумма ед_изм="руб.">275000,00</сумма>
    </вкладчик>
  </вклад>
  <вклад код="тн261">
    <вид>сберегательный</вид>
    <процент>8,19</процент>
    <вкладчик инн="762248659912">
      <фio>Бутенко Алексей Викторович</фio>
      <сумма ед_изм="руб.">680000,00</сумма>
    </вкладчик>
  </вклад>
</банк>
</банковские_услуги>

```

Напишем исходный код процедур выполнения запросов к XML-документу:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Xml.Linq;
using System.Xml.XPath;

namespace ЛР_2_3
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.Title = "Выполнение запросов LINQ к XML-документу";
            // Загрузка XML-документа
            XDocument xmlDoc = XDocument.Load("bank.xml");
            Console.WriteLine("\t\tРезультаты запросов LINQ к XML-документу");
            // Вызовы процедур, выполняющих запросы к XML-документу
            GetTypeSum(xmlDoc);
            GetNumDep(xmlDoc);
            GetDepMore(xmlDoc);
            GetListDep(xmlDoc);
            GetMoreOne(xmlDoc);
            Console.ReadLine();
        }
        // Вид и сумма вкладов с процентной ставкой более 10%
    }
}

```

```

static void GetTypeSum(XDocument doc)
{
    var deposit = from v in doc.Descendants("вклад")
                   where Convert.ToDouble(v.Element("процент").Value) > 10
                   select v;
    Console.WriteLine("\n1. Вид и сумма вкладов с процентной ставкой более 10%:");
    foreach (var d in deposit)
    {
        Console.WriteLine("- " + d.Element("вид").Value + ": ");
        var depositor = from v in d.Descendants("вкладчик")
                         select v;
        double sumDep = 0;
        foreach (var dep in depositor)
        {
            sumDep += Convert.ToDouble(dep.Element("сумма").Value);
        }
        Console.WriteLine((string.Format("{0:0.00}", sumDep) + " руб.));
    }
}
// Число вкладов, по которым процентная ставка составляет от 5 до 12%
static void GetNumDep(XDocument doc)
{
    var deposit = from v in doc.Descendants("вклад")
                   where Convert.ToDouble(v.Element("процент").Value) >= 5 &&
                        Convert.ToDouble(v.Element("процент").Value) <= 12
                   select v;
    int numDep = deposit.Count();
    Console.WriteLine("\n2. Число вкладов, по которым процентная ставка составляет от
        5 до 12%: " +
        numDep);
}
// Данные по вкладам размером более 150000,00 руб., сгруппированные по клиентам
static void GetDepMore(XDocument doc)
{
    var groups = from v in doc.Descendants("вклад")
                  from d in v.Elements("вкладчик")
                  where Convert.ToDouble(d.Element("сумма").Value) > 150000
                  group v by d.Element("фио").Value;
    Console.WriteLine("\n3. Данные по вкладам размером более 150000,00 руб.,
        сгруппированные по клиентам:");
    // Цикл для выбора каждой группы group из списка групп groups
    foreach (var group in groups)
    {
        Console.WriteLine("Вкладчик: " + group.Key);
        // Цикл для выбора каждого элемента elem группы group
        foreach (var elem in group)
        {
            Console.WriteLine("- вклад \"" + elem.Element("вид").Value + "\" +
                " (код " + elem.Attribute("код").Value + ") " +
                " в банке " + (elem.Parent.Attribute("название")).Value +
                " (БИК " + elem.Parent.Attribute("бик").Value + ") " +
                "\n под " + elem.Element("процент").Value + " процентов годовых");
        }
    }
}
// Список вкладов с указанием данных по вкладчику
static void GetListDep(XDocument doc)
{
    XElement customers = new XElement("клиенты_банков",
        new XElement("клиент",
            new XAttribute("инн", "374981357822"),
            new XElement("фио", "Воробьев Сергей Борисович"),
            new XElement("дата_рождения", "21.04.1979"),
            new XElement("адрес", "ул. Севастопольская, д. 5, кв. 48")
        ),

```

```

        new XElement("клиент",
            new XAttribute("инн", "761594352278"),
            new XElement("фio", "Прокопьев Антон Сергеевич"),
            new XElement("дата_рождения", "13.05.1985"),
            new XElement("адрес", "ул. Суздальская, д. 36, кв. 15")
        ),
        new XElement("клиент",
            new XAttribute("инн", "762248659912"),
            new XElement("фio", "Бутенко Алексей Викторович"),
            new XElement("дата_рождения", "26.11.2001"),
            new XElement("адрес", "ул. Промышленная, д. 1, кв. 55")
        ),
        new XElement("клиент",
            new XAttribute("инн", "117856842341"),
            new XElement("фio", "Копылов Семен Андреевич"),
            new XElement("дата_рождения", "02.12.1995"),
            new XElement("адрес", "ул. Промышленная, д. 1, кв. 55")
        )
    );
    var list = from b in doc.Descendants("банк")
               from v in b.Elements("вклад")
               from d in v.Elements("вкладчик")
               join a in customers.Elements("клиент")
                   on d.Attribute("инн").Value equals a.Attribute("инн").Value
               select new
               {
                   bank = b.Attribute("название").Value,
                   bik = b.Attribute("бик").Value,
                   dep = v.Element("вид").Value,
                   inn = a.Attribute("инн").Value,
                   fio = a.Element("фio").Value,
                   date = a.Element("дата_рождения").Value,
                   addr = a.Element("адрес").Value
               };
    Console.WriteLine("\n4. Список вкладов с указанием данных вкладчиков:");
    foreach (var l in list)
    {
        Console.WriteLine("- вклад \" " + l.dep + "\" " +
            "в банке \" " + l.bank + "\" " +
            "(БИК \" " + l.bik + "\" " +
            "\n на имя \" " + l.fio +
            " (ИНН \" " + l.inn + "\" " +
            "\n дата рождения: \" " + l.date +
            "\", адрес: \" " + l.addr);
    }
}
// Клиенты, сделавшие более одного вклада
static void GetMoreOne(XDocument doc)
{
    var more = doc.XPathSelectElements("//вклад/вкладчик/фio");
    Console.WriteLine("\n5. Клиенты, сделавшие более одного вклада:");
    var set = new HashSet<string>();
    var moreOne = new HashSet<string>();
    foreach (var m in more)
    {
        if (!set.Add(m.Value))
            moreOne.Add(m.Value);
    }
    foreach (var m in moreOne)
    {
        Console.WriteLine("- \" " + m);
    }
}
}
}
}

```



### 3.3 Результат работы консольного приложения

После запуска программы получен результат согласно рисунка 1.

```
Выполнение запросов LINQ × + ▾ - □ ×

Результаты запросов LINQ к XML-документу

1. Вид и сумма вкладов с процентной ставкой более 10%:
- срочный: 1105000,00 руб.
- накопительный: 1250000,00 руб.

2. Число вкладов, по которым процентная ставка составляет от 5 до 12%: 3

3. Данные по вкладам размером более 150000,00 руб., сгруппированные по клиентам:
Вкладчик: Прокопьев Антон Сергеевич
- вклад "срочный" (код сб137) в банке Сбербанк (БИК 044525225)
  под 11,6 процентов годовых
- вклад "до востребования" (код тн175) в банке Тинькофф (БИК 044525974)
  под 5,16 процентов годовых
Вкладчик: Бутенко Алексей Викторович
- вклад "срочный" (код сб137) в банке Сбербанк (БИК 044525225)
  под 11,6 процентов годовых
- вклад "социальный" (код сб318) в банке Сбербанк (БИК 044525225)
  под 1,95 процентов годовых
- вклад "сберегательный" (код тн261) в банке Тинькофф (БИК 044525974)
  под 8,19 процентов годовых
Вкладчик: Копылов Семен Андреевич
- вклад "накопительный" (код сб265) в банке Сбербанк (БИК 044525225)
  под 15,3 процентов годовых

4. Список вкладов с указанием данных вкладчиков:
- вклад "срочный" в банке "Сбербанк" (БИК 044525225)
  на имя Воробьев Сергей Борисович (ИНН 374981357822)
  дата рождения: 21.04.1979, адрес: ул. Севастопольская, д. 5, кв. 48
- вклад "срочный" в банке "Сбербанк" (БИК 044525225)
  на имя Прокопьев Антон Сергеевич (ИНН 761594352278)
  дата рождения: 13.05.1985, адрес: ул. Суздальская, д. 36, кв. 15
- вклад "срочный" в банке "Сбербанк" (БИК 044525225)
  на имя Бутенко Алексей Викторович (ИНН 762248659912)
  дата рождения: 26.11.2001, адрес: ул. Промышленная, д. 1, кв. 55
- вклад "накопительный" в банке "Сбербанк" (БИК 044525225)
  на имя Копылов Семен Андреевич (ИНН 117856842341)
  дата рождения: 02.12.1995, адрес: ул. Промышленная, д. 1, кв. 55
- вклад "социальный" в банке "Сбербанк" (БИК 044525225)
  на имя Воробьев Сергей Борисович (ИНН 374981357822)
  дата рождения: 21.04.1979, адрес: ул. Севастопольская, д. 5, кв. 48
- вклад "социальный" в банке "Сбербанк" (БИК 044525225)
  на имя Бутенко Алексей Викторович (ИНН 762248659912)
  дата рождения: 26.11.2001, адрес: ул. Промышленная, д. 1, кв. 55
- вклад "до востребования" в банке "Тинькофф" (БИК 044525974)
  на имя Воробьев Сергей Борисович (ИНН 374981357822)
  дата рождения: 21.04.1979, адрес: ул. Севастопольская, д. 5, кв. 48
- вклад "до востребования" в банке "Тинькофф" (БИК 044525974)
  на имя Прокопьев Антон Сергеевич (ИНН 761594352278)
  дата рождения: 13.05.1985, адрес: ул. Суздальская, д. 36, кв. 15
- вклад "сберегательный" в банке "Тинькофф" (БИК 044525974)
  на имя Бутенко Алексей Викторович (ИНН 762248659912)
  дата рождения: 26.11.2001, адрес: ул. Промышленная, д. 1, кв. 55

5. Клиенты, сделавшие более одного вклада:
- Воробьев Сергей Борисович
- Бутенко Алексей Викторович
```

Рисунок 1 – Результат работы консольного приложения