

**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**ITMO University**

**ЛАБОРАТОРНАЯ РАБОТА №2**

**По дисциплине** Объектно-ориентированное программирование

**Тема работы** Создание и использование размерных типов данных

**Обучающийся** Крестьянова Елизавета Федоровна

**Факультет** факультет инфокоммуникационных технологий

**Группа** К3223

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и  
системы связи

**Образовательная программа** Программирование в  
инфокоммуникационных системах

<b>Обучающийся</b>	_____	_____	<u>Крестьянова Е.Ф.</u>
	(дата)	(подпись)	(Ф.И.О.)
<b>Руководитель</b>	_____	_____	<u>Иванов С.Е.</u>
	(дата)	(подпись)	(Ф.И.О.)

## СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ .....	3
1 Упражнение №1. Создание перечисления .....	4
2 Упражнение №2. Создание и использование структуры .....	6
3 Упражнение №3. Реализация структуры Distance .....	10
ЗАКЛЮЧЕНИЕ .....	12

## **ВВЕДЕНИЕ**

В данном отчёте представлено выполнение лабораторной работы по дисциплине «Объектно-ориентированное программирование».

Цель данной работы - изучение размерных типов данных и приобретение навыков работы со структурными типами.

## 1 УПРАЖНЕНИЕ №1. СОЗДАНИЕ ПЕРЕЧИСЛЕНИЯ

В этом упражнении стоит задача создания перечисления для представления различных типов банковских счетов. Данное перечисление будет использоваться для создания двух переменных, которым присваиваются значения Checking и Deposit. Эти значения должны быть выведены.

Был создан новый проект BankAccount с классом Enum. Было создано перечисление AccountType со значениями Checking и Deposit. Они были присвоены переменным goldAccount и platinumAccount, которые затем были выведены в консоль.

Код программы можно увидеть на рисунке 1.1.

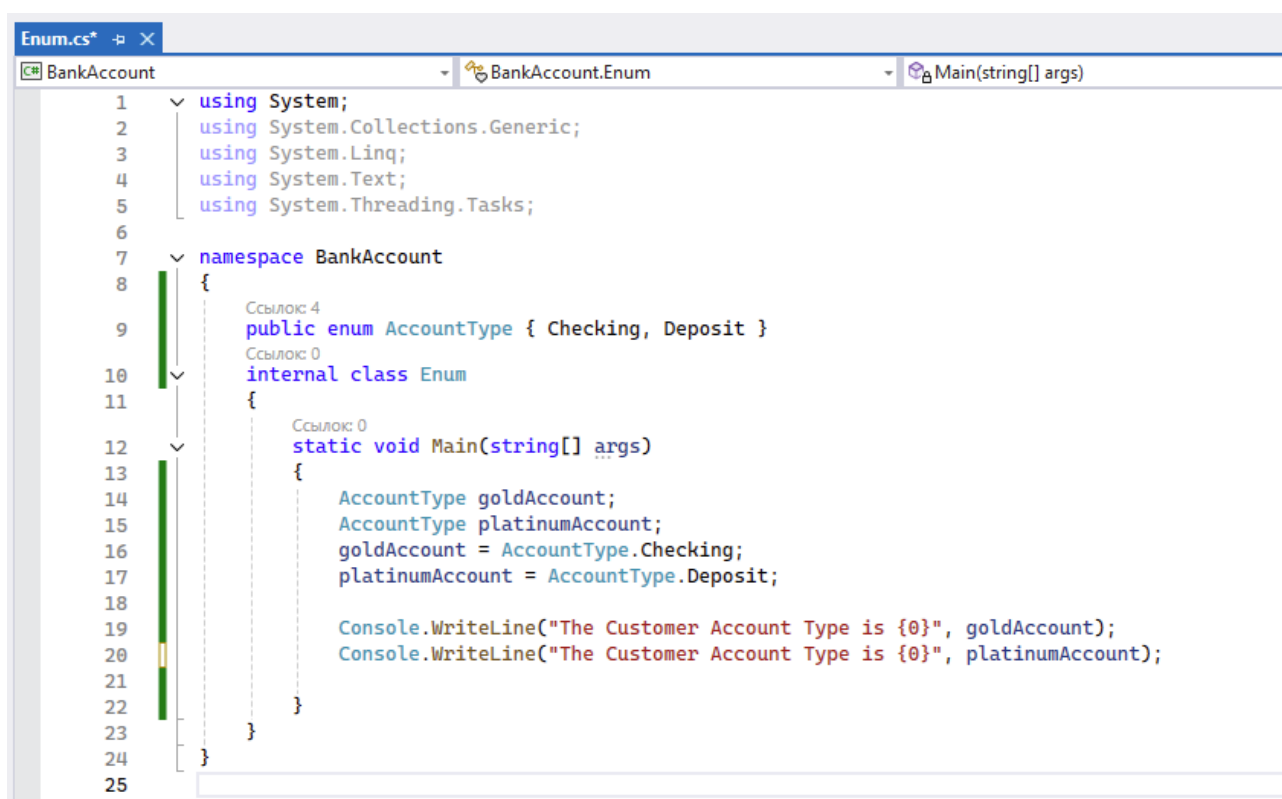


Рисунок 1.1 — Упр №1: Код программы

Вывод программы можно увидеть на рисунке 1.2.

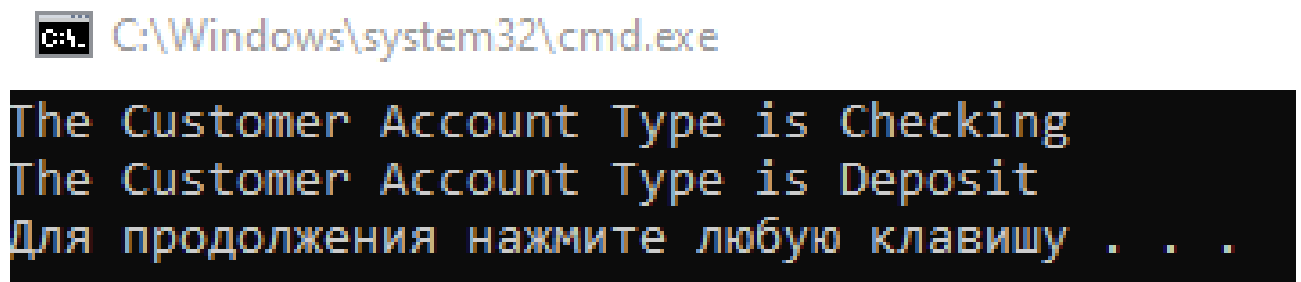


Рисунок 1.2 — Упр №2: Вывод программы

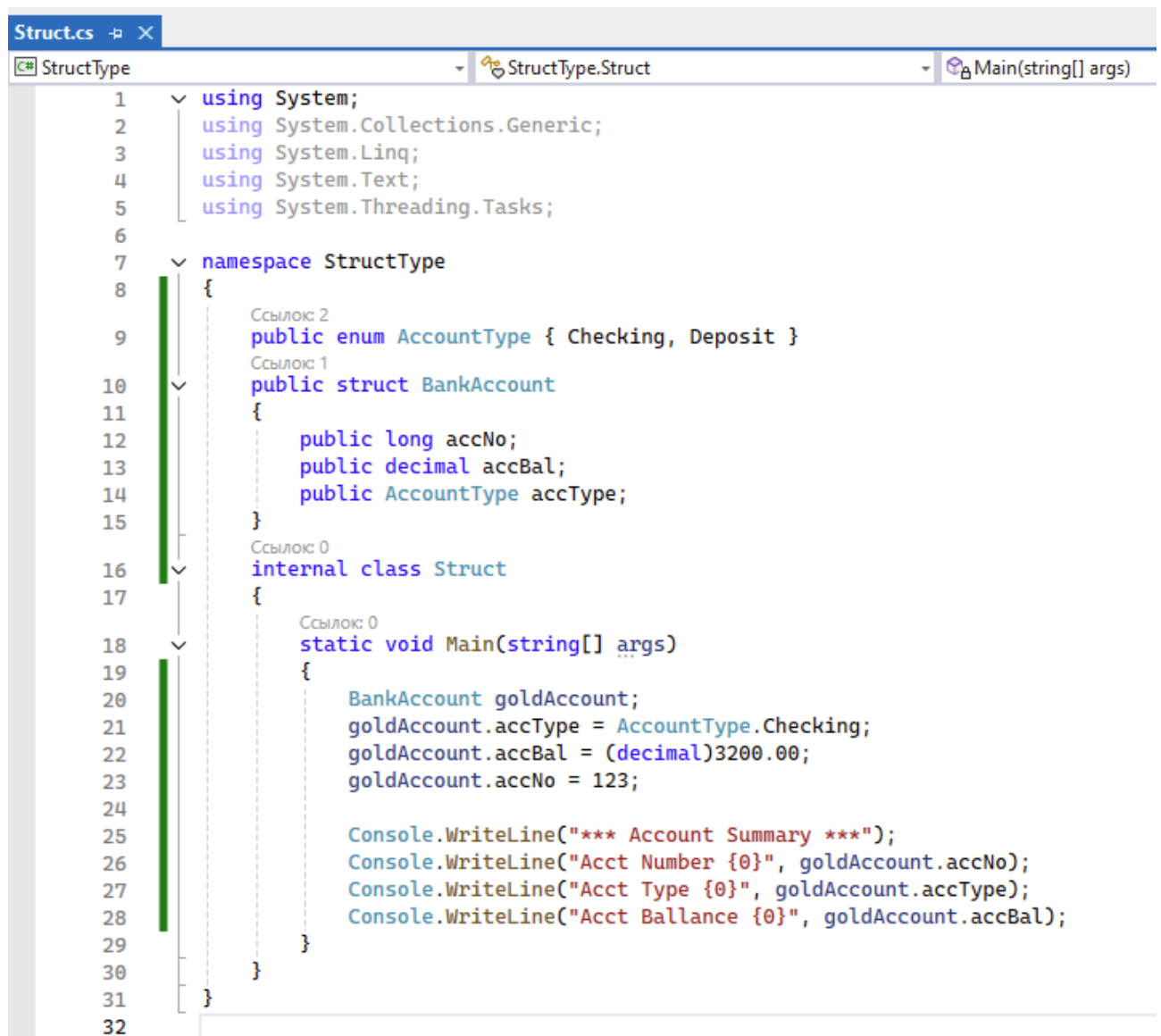
## 2 УПРАЖНЕНИЕ №2. СОЗДАНИЕ И ИСПОЛЬЗОВАНИЕ СТРУКТУРЫ

Это упражнение нацелено на создание структуры, которую можно использовать для представления банковских счетов. Для хранения номеров счетов (тип данных `long`), балансов счетов (тип данных `decimal`) и типов счетов (перечисление из упражнения №1) будут использованы переменные. Должен быть создан переменный тип структуры, заполненный данными и выводимый в консоль.

Был создан новый проект `StructType` с классом `Struct`. Как и в предыдущем упражнении, было создано перечисление `AccountType`. Так же была создана структура `BankAccount` с переменными номера счёта, баланса счёта и типа счёта, которому присваиваются значения из перечисления `AccountType`.

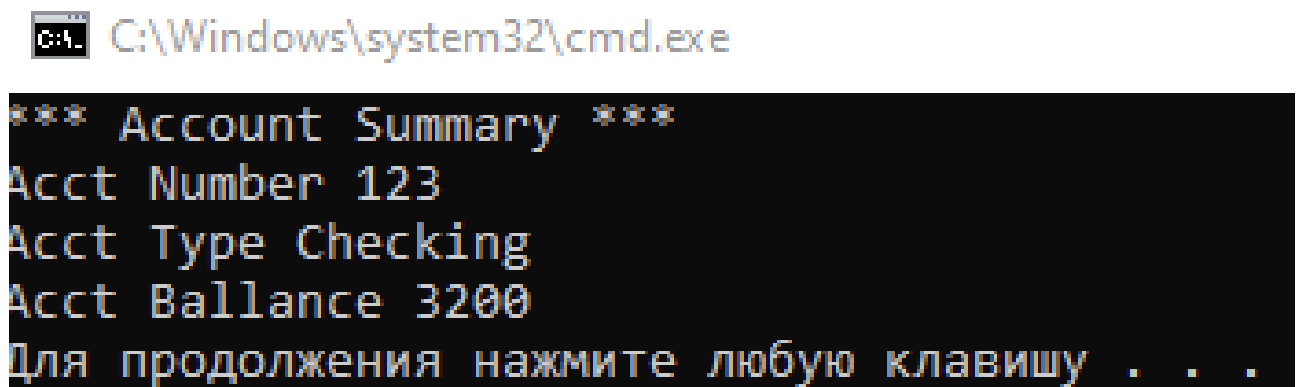
В методе `Main` была создана переменная `goldAccount` со структурой `BankAccount`. Ей были переданы данные к каждой переменной, и затем вся информация о ней была выведена в консоль.

Код и вывод программы можно увидеть на рисунках 2.1 и 2.2 соответственно.



```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace StructType
8  {
9      Ссылка: 2
10     public enum AccountType { Checking, Deposit }
11     Ссылка: 1
12     public struct BankAccount
13     {
14         public long accNo;
15         public decimal accBal;
16         public AccountType accType;
17     }
18     Ссылка: 0
19     internal class Struct
20     {
21         Ссылка: 0
22         static void Main(string[] args)
23         {
24             BankAccount goldAccount;
25             goldAccount.accType = AccountType.Checking;
26             goldAccount.accBal = (decimal)3200.00;
27             goldAccount.accNo = 123;
28
29             Console.WriteLine("*** Account Summary ***");
30             Console.WriteLine("Acct Number {0}", goldAccount.accNo);
31             Console.WriteLine("Acct Type {0}", goldAccount.accType);
32             Console.WriteLine("Acct Ballance {0}", goldAccount.accBal);
33         }
34     }
35 }
```

Рисунок 2.1 — Упр №2: Код программы



```
C:\Windows\system32\cmd.exe
*** Account Summary ***
Acct Number 123
Acct Type Checking
Acct Ballance 3200
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.2 — Упр №2: Вывод программы

Затем в программу был добавлен ввод пользователя, который записывается в переменную `accNo` аккаунта. Также был добавлен обработчик исключительных ситуаций, необходимый для проверки правильности ввода.

Код программы и её работа представлены на рисунках 2.3 и 2.4 соответственно.

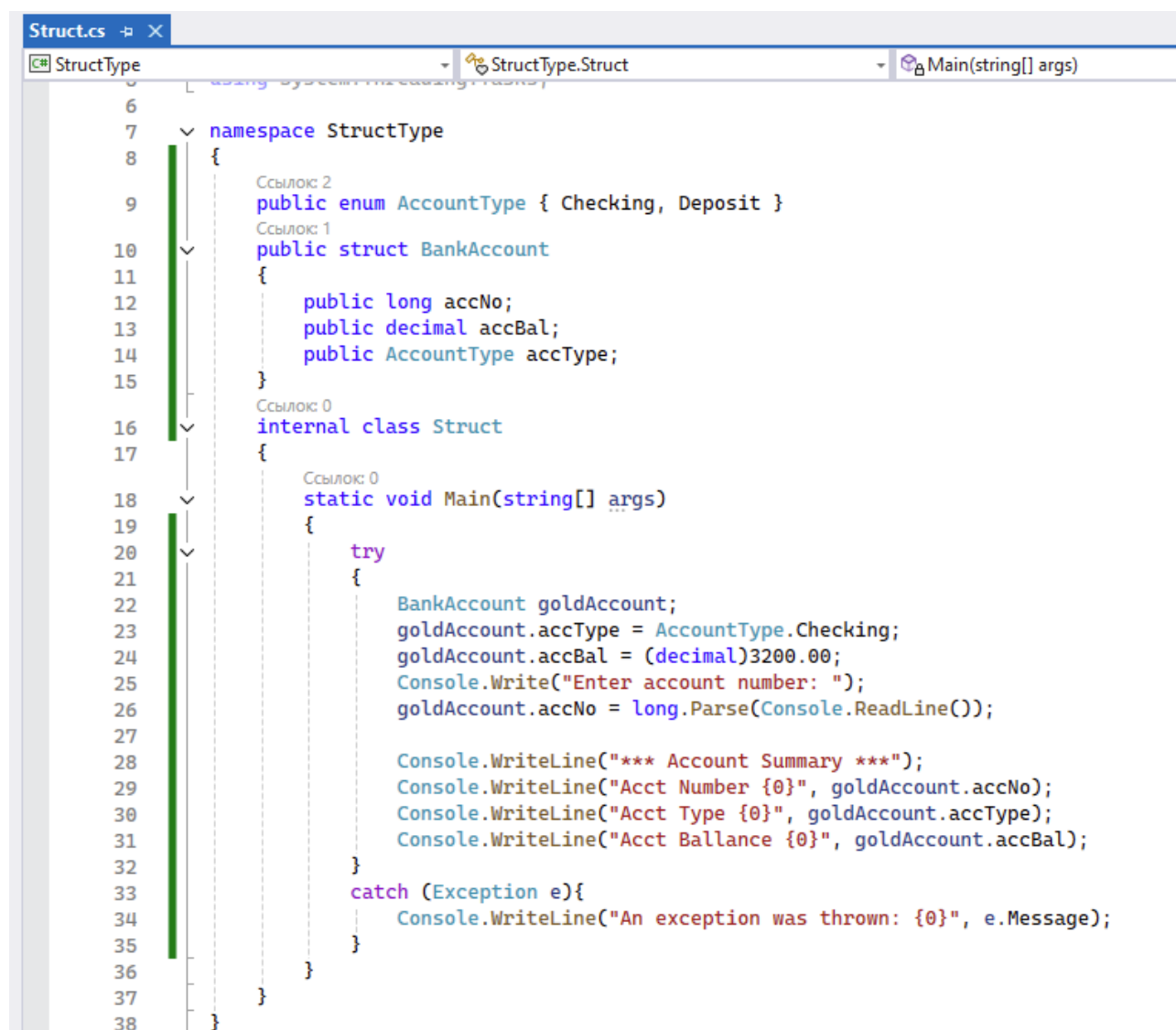


Рисунок 2.3 — Упр №2: Код программы, принимающей ввод пользователя



---

C:\Windows\system32\cmd.exe

```
Enter account number: 1111
*** Account Summary ***
Acct Number 1111
Acct Type Checking
Acct Ballance 3200
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2.4 — Упр №2: Пользовательский ввод

### **3 УПРАЖНЕНИЕ №3. РЕАЛИЗАЦИЯ СТРУКТУРЫ DISTANCE**

Данное упражнение требует создания структуры Distance, определяющей длину в английской системе мер.

Был создан новый проект Distance. В нём была указана структура Distance с целочисленными переменными feet и inches. Пользователь задаёт значения футов и дюймов двум переменным dist1 и dist2. Затем их значения складываются в переменную dist3. После этого сумма пересчитывается: излишние дюймы переходят в футы.

Код программы и пример её вывода приведены на рисунках 3.1 и 3.2 соответственно.

```
Program.cs  + X
Distance
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Distance
{
    public struct Distance
    {
        public int feet;
        public int inches;
    }

    internal class Program
    {
        static void Main(string[] args)
        {
            try
            {
                Distance dist1, dist2, dist3;

                Console.WriteLine("Введите значение футов первой переменной: ");
                dist1.feet = int.Parse(Console.ReadLine());
                Console.WriteLine("Введите значение дюймов первой переменной: ");
                dist1.inches = int.Parse(Console.ReadLine());

                Console.WriteLine("Введите значение футов второй переменной: ");
                dist2.feet = int.Parse(Console.ReadLine());
                Console.WriteLine("Введите значение дюймов второй переменной: ");
                dist2.inches = int.Parse(Console.ReadLine());

                dist3.inches = dist1.inches + dist2.inches;
                dist3.feet = dist1.feet + dist2.feet;

                dist3.feet += (int)(dist3.inches / 12);
                dist3.inches = dist3.inches % 12;

                Console.WriteLine("Сумма переменных: {0} ' - {1}''", dist3.feet, dist3.inches);
            }
            catch (Exception e)
            {
                Console.WriteLine("An exception was thrown: {0}", e);
            }
        }
    }
}
```

Рисунок 3.1 — Упр №3: Код программы

```
Введите значение футов первой переменной: 3
Введите значение дюймов первой переменной: 7
Введите значение футов второй переменной: 1
Введите значение дюймов второй переменной: 8
Сумма переменных: 5 ' - 3''
Для продолжения нажмите любую клавишу . . .
```

Рисунок 3.2 — Упр №3: Вывод программы

## **ЗАКЛЮЧЕНИЕ**

При выполнении лабораторной работы были созданы три программы, в которых были созданы различные структурные типы. Их данные были заполнялись как самой программой, так и пользовательским вводом.

Цель изучения размерных типов данных и приобретения навыков работы со структурными типами была выполнена.