

**Министерство науки и высшего образования Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  
**«Национальный исследовательский университет ИТМО»**  
**(Университет ИТМО)**

Факультет инфокоммуникационных технологий

Отчет по дисциплине: **«Объектно-ориентированное программирование»**  
Лабораторная работа 2. "Создание и использование размерных типов данных"

Выполнил: Абдулов  
Илья Александрович

Группа: К3221

Проверил: Васильев  
Сергей Юрьевич

Санкт-Петербург  
2023

## Содержание

Введение . . . . .	3
1 Создание и использование размерных типов данных . . . . .	4
1.1 Упражнение 1. Создание перечисления . . . . .	4
1.2 Упражнение 2. Создание и использование структуры . .	5
1.3 Упражнение 3. Реализация структуры Distance . . . . .	7
Заключение . . . . .	8

## Введение

Целью работы является изучение размерных типов данных и приобретение навыков работы со структурными типами. Для достижения поставленной цели необходимо решить следующие упражнения:

1. Создать перечисление;
2. Создать и использовать структуру;
3. Реализовать структуру Distance.

Лабораторная выполняется на компьютере под macOS, поэтому некоторые этапы и инструменты выполнения работы будут отличаться от рекомендуемых в Windows.

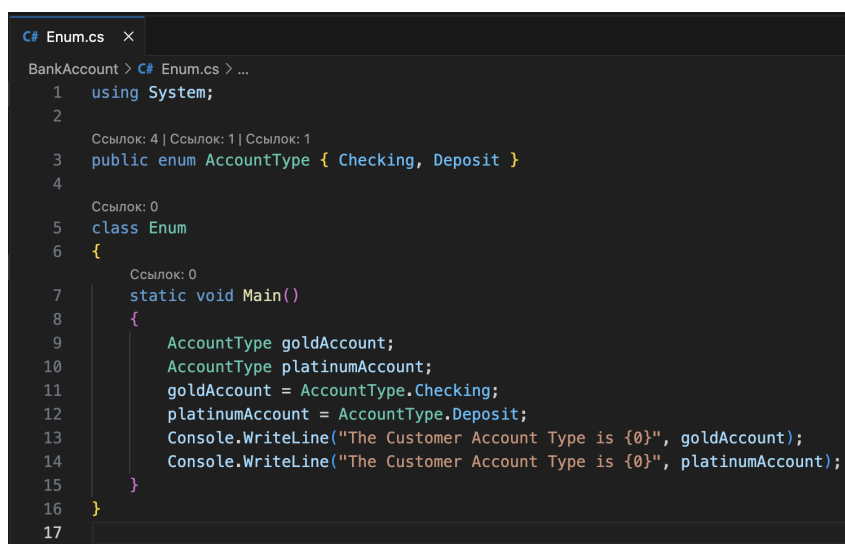
# 1 Создание и использование размерных типов данных

В данном разделе изучаются размерные типы данных и приобретаются навыки работы со структурными типами.

## 1.1 Упражнение 1. Создание перечисления

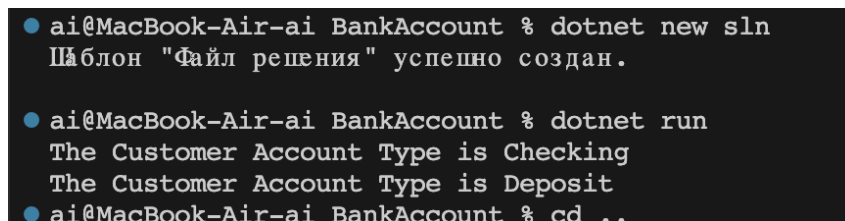
Создадим новый проект BankAccount. Переименуем стандартный файл Program.cs на Enum.cs, перед описанием класса добавим перечисление AccountType. В методе Main() объявим две переменные и присвоим им значения типа AccountType (Рисунок 1.1).

Программа выведет на экран значения Checking и Deposit перечисления AccountType (Рисунок 1.2).



```
C# Enum.cs X
BankAccount > C# Enum.cs > ...
1  using System;
2
3  Ссылка: 4 | Ссылка: 1 | Ссылка: 1
   public enum AccountType { Checking, Deposit }
4
5  Ссылка: 0
   class Enum
6  {
7      Ссылка: 0
       static void Main()
8      {
9          AccountType goldAccount;
10         AccountType platinumAccount;
11         goldAccount = AccountType.Checking;
12         platinumAccount = AccountType.Deposit;
13         Console.WriteLine("The Customer Account Type is {0}", goldAccount);
14         Console.WriteLine("The Customer Account Type is {0}", platinumAccount);
15     }
16 }
17
```

Рисунок 1.1 — Код программы с перечислением AccountType



```
ai@MacBook-Air-ai BankAccount % dotnet new sln
Шаблон "Файл решения" успешно создан.

ai@MacBook-Air-ai BankAccount % dotnet run
The Customer Account Type is Checking
The Customer Account Type is Deposit

ai@MacBook-Air-ai BankAccount % cd ..
```

Рисунок 1.2 — Вывод перечисления

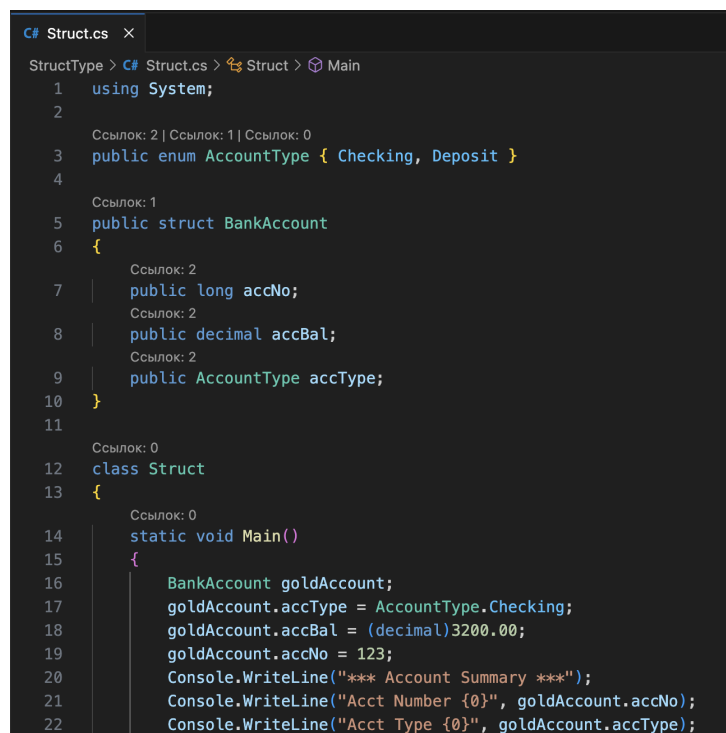
## 1.2 Упражнение 2. Создание и использование структуры

В этом упражнении мы создадим структуру, которую можно использовать для представления банковских счетов.

Создадим новый проект C# с названием StructType.sln в директории /Lab02/StructType и переименуем файл Program.cs на Struct.cs.

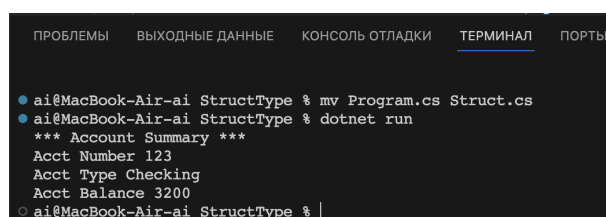
Добавим перечисление AccountType. Создадим public структуру BankAccount, которая будет иметь три переменные: accNo типа long, accBal типа decimal, accType типа перечисления AccountType.

В методе Main() объявим переменную типа BankAccount и присвоим значения полям accNo, accBal и accType (Рисунок 1.3). Выведем значения структуры в консоль (Рисунок 1.4).



```
C# Struct.cs X
StructType > C# Struct.cs > Struct > Main
1 using System;
2
3 Ссылка: 2 | Ссылка: 1 | Ссылка: 0
public enum AccountType { Checking, Deposit }
4
5 Ссылка: 1
public struct BankAccount
6 {
7     Ссылка: 2
    public long accNo;
8     Ссылка: 2
    public decimal accBal;
9     Ссылка: 2
    public AccountType accType;
10 }
11
12 Ссылка: 0
class Struct
13 {
14     Ссылка: 0
    static void Main()
15     {
16         BankAccount goldAccount;
17         goldAccount.accType = AccountType.Checking;
18         goldAccount.accBal = (decimal)3200.00;
19         goldAccount.accNo = 123;
20         Console.WriteLine("*** Account Summary ***");
21         Console.WriteLine("Acct Number {0}", goldAccount.accNo);
22         Console.WriteLine("Acct Type {0}", goldAccount.accType);
```

Рисунок 1.3 — Код Struct.cs структура BankAccount



```
ПРОБЛЕМЫ Выходные данные КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ
ai@MacBook-Air-ai StructType % mv Program.cs Struct.cs
ai@MacBook-Air-ai StructType % dotnet run
*** Account Summary ***
Acct Number 123
Acct Type Checking
Acct Balance 3200
ai@MacBook-Air-ai StructType %
```

Рисунок 1.4 — Вывод Struct.cs

Добавим возможность ввода переменной структуры `accNo`. При вводе будем преобразовывать его из типа `string` в тип `long`, используя метод `Long.Parse`. Рассмотрим работу измененной программы с элементом ввода от пользователя (Рисунок 1.5).

```
● ai@MacBook-Air-ai StructType % dotnet run
/Users/ai/Developer/itmo-uni/sem3/OOP/labs/
ение NULL, для параметра "s" в "long long.I
j]
Enter account number: 9384
*** Account Summary ***
Acct Number 9384
Acct Type Checking
Acct Balance 3200
○ ai@MacBook-Air-ai StructType % □
```

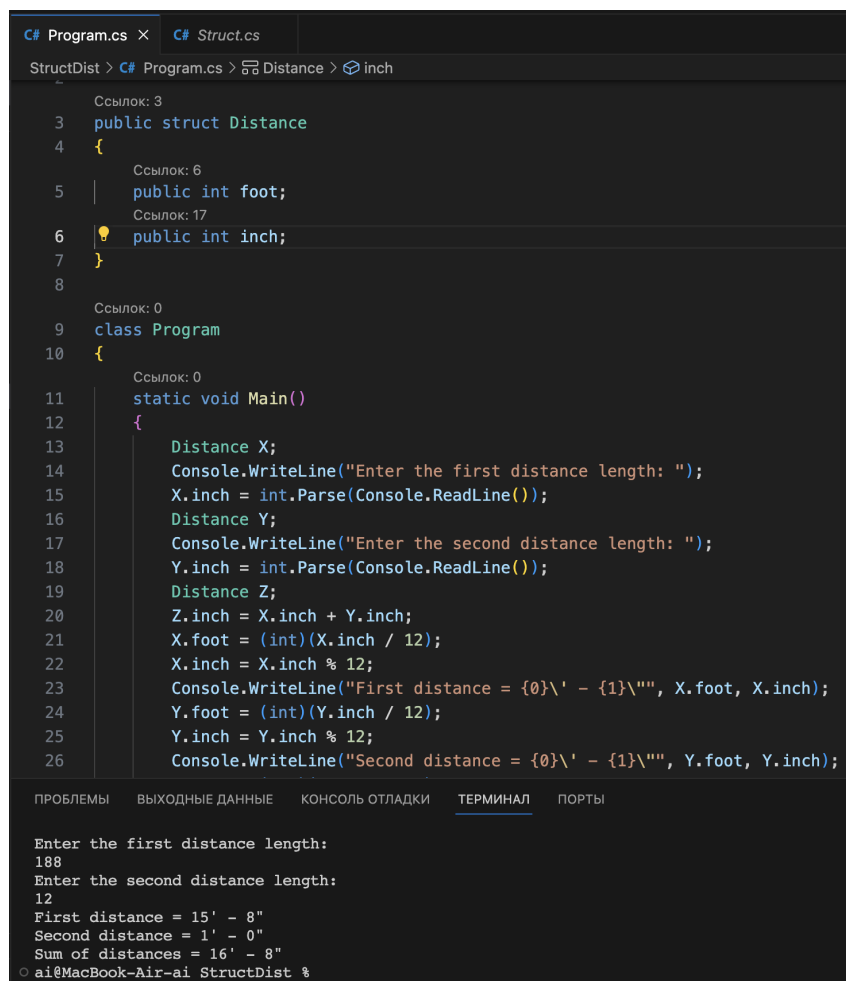
Рисунок 1.5 — Получение и вывод `accNo`

### 1.3 Упражнение 3. Реализация структуры Distance

В этом задании создадим структуру Distance, определяющую длину в английской системе мер.

Создадим новый проект StructDist. В файле Program.cs до объявления класса Program создадим public структуру Distance в которой будет две public переменные типа int: foot и inch.

В методе Main() класса Program инициализируем две переменные типа Distance с помощью ввода пользователя. Третья переменная типа Distance будет равна суммы двух введенных. Пересчитаем значения foot и inch переменных с учетом того, что один фут равен 12 дюймам. Представим результаты длин в английской системе мер в консоли. Код и вывод программы представлен ниже (Рисунок 1.6).



```
C# Program.cs X C# Struct.cs
StructDist > C# Program.cs > Distance > inch

3   Ссылка: 3
4   public struct Distance
5   {
6       Ссылка: 6
7       public int foot;
8       Ссылка: 17
9       public int inch;
10  }
11
12  Ссылка: 0
13  class Program
14  {
15      Ссылка: 0
16      static void Main()
17      {
18          Distance X;
19          Console.WriteLine("Enter the first distance length: ");
20          X.inch = int.Parse(Console.ReadLine());
21          Distance Y;
22          Console.WriteLine("Enter the second distance length: ");
23          Y.inch = int.Parse(Console.ReadLine());
24          Distance Z;
25          Z.inch = X.inch + Y.inch;
26          X.foot = (int)(X.inch / 12);
27          X.inch = X.inch % 12;
28          Console.WriteLine("First distance = {0}' - {1}\"", X.foot, X.inch);
29          Y.foot = (int)(Y.inch / 12);
30          Y.inch = Y.inch % 12;
31          Console.WriteLine("Second distance = {0}' - {1}\"", Y.foot, Y.inch);
32          Z.foot = (int)(Z.inch / 12);
33          Z.inch = Z.inch % 12;
34          Console.WriteLine("Sum of distances = {0}' - {1}\"", Z.foot, Z.inch);
35      }
36  }

ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ  ПОРТЫ

Enter the first distance length:
188
Enter the second distance length:
12
First distance = 15' - 8"
Second distance = 1' - 0"
Sum of distances = 16' - 8"
o ai@MacBook-Air-ai StructDist %
```

Рисунок 1.6 — Реализация структуры Distance

## Заключение

В результате проделанной лабораторной работы были изучены размерные типы данных и приобретены навыки работы со структурными типами. Для достижения поставленной цели необходимо были решены следующие задачи:

1. Создано перечисление для представления банковских счетов;
2. Созданы и использованы тип данных структура;
3. Реализована структура Distance.