# **7 Структурная обработка исключений в .NET, проектирование собственных типов исключений**

Задание 1. Создать исключение InvalidAccountNumberException, если номер банковского счета состоит не из 10 цифр. Класс BankAccount должен содержать метод ValidateAccount(string accountNumber), проверяющий корректность номера счета.

Листинг программы:

using System;

namespace E

{

public class InvalidAccountNumberException : Exception

{

public InvalidAccountNumberException()

: base("Некорректный номер банковского счета.") { }

public InvalidAccountNumberException(string m)

: base(m) { }

public InvalidAccountNumberException(string m, Exception innerException)

: base(m, innerException) { }

}

public class BankAccount

{

public void ValidateAccount(string accountNumber)

{

if (accountNumber.Length != 10 || !long.TryParse(accountNumber, out \_))

{

throw new InvalidAccountNumberException($"Номер счета должен содержать ровно 10 цифр. Вы ввели \"{accountNumber}\"."); } } }

class BAV

{

static void Main()

{

BankAccount a = new BankAccount();

try

{

Console.Write("Введите номер банковского счета: ");

string accountNumber = Console.ReadLine();

a.ValidateAccount(accountNumber);

Console.WriteLine("Номер счета корректен!");

}

catch (InvalidAccountNumberException e)

{

Console.WriteLine($"{e.Message}");

}

}

}

}

Таблица 7.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1234567890 | Номер счета корректен! |

Анализ результатов:

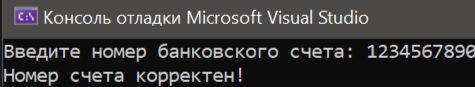


Рисунок 7.1 – Результат работы программы