**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,**

**СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

Факультет «Информационных систем и технологий»

Кафедра «Интеллектуальных систем автоматизации и управления»

|  |  |
| --- | --- |
| Направление подготовки: | 09.03.02 - Информационные системы и технологии |
| Направленность (профиль): | Системное и прикладное программирование информационных  систем |

**ЛАБОРАТОРНАЯ РАБОТА № 1**

по дисциплине:

|  |
| --- |
| **Введение в программную инженерию** |

на тему:

|  |
| --- |
| **«СОЗДАНИЕ КЛАССОВ И ОБЪЕКТОВ»** |

Вариант № 8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Выполнил студент группы | | ИСТ-261 |
|  |  | Филатов М.Ю. | | |
|  |  | *Фамилия И. О.* | | |
|  |  | Руководитель | ассистент кафедры ИСАУ | |
| *оценка* |  |  | *уч. степень, уч. звание* | |
|  |  | Шабанов А.П. | | |
| *дата, подпись* |  | *Фамилия И. О.* | | |

**Цель работы**

На практике ознакомиться с объявлением классов и различными вариантами создания объектов на языке программирования C#, а также научиться работать с коллекциями объектов.

**Постановка задачи**

1. В интегрированной среде разработки MS Visual Studio 2022 создать консольное приложение. В данном приложении создать класс «Почтовое отправление» и объекты, являющиеся экземплярами этого класса. Свойства класса «Почтовое отправление»: индекс, город, улица, дом, корпус, квартира, а также тело письма (сообщение). У класса должны быть конструктор без параметров (пустой конструктор), конструктор с параметрами, копирующий конструктор и метод клонирования объекта. Также класс должен содержать метод Valid(), выполняющий проверку заполнения полей и свойство «Адрес», выводящее адрес в виде строки.
2. Инициализировать четыре экземпляра класса «Почтовое отправление». Первый объект инициализируется информацией, полученной от пользователя из командной строки. Второй объект инициализируется программно. Третий объект получается путем копирования первого объекта с помощью применения копирующего конструктора, четвертый – путем клонирования второго объекта.
3. Создать коллекцию из пяти объектов. Сделать вывод информации об объектах в цикле foreach. Удалить два объекта по индексам 1 и 3. Вывести информацию об оставшихся объектах в цикле foreach.

**Ход работы**

Создаётся класс Mail со свойствами Index (индекс), City (город), Street (улица), House (дом), Building (корпус), Flat (квартира), Message (сообщение) и Address (адрес). Все свойства, кроме свойства Address, имеют модификатор доступа private для того, чтобы не нарушать один из принципов объектно-ориентированного программирования - инкапсуляцию. Свойство Address, выводящее на экран адрес в виде строки, имеет модификатор доступа public, так как в последующем напрямую будет использоваться в программе. Согласно заданию, объявляется конструктор без параметров, конструктор с параметрами, копирующий конструктор и клонирующий метод. В теле класса создаётся метод Valid, возвращающий либо true, либо false, в зависимости от того, были ли переданы свойствам «непустые» значения или нет.

**Эксперимент 1**

Далее, в методе Main объявляются четыре экземпляра класса Mail. Первый объект инициализируется информацией, полученной от пользователя из командной строки (см. рис. 1), второй объект инициализируется, получая значения для свойств через конструктор с параметрами, третий объект – копия первого, а четвёртый объект получается путём клонирования второго экземпляра класса Mail.

При создании первого объекта для проверки верного заполнения аргументов пользователем и избежания ошибки применяется условный тернарный оператор – укороченный (свёрнутый) условный оператор if. В том случае, если пользователь введёт меньшее количество аргументов, чем требуется, остальные свойства примут пустые значения (null).

Для вывода информации об объектах используется статический метод PrintMail, принимающий в качестве аргумента экземпляр класса Mail: выводится адрес и выполняется «проверка» (вызывается метод Valid()) на то, что свойства не являются пустыми значениями.

**class** **Mail**

{

*// Properties*

**private** String? Index { **get**; **set**; }

**private** String? City { **get**; **set**; }

**private** String? Street { **get**; **set**; }

**private** String? House { **get**; **set**; }

**public** String? Building { **get**; **set**; }

**private** String? Flat { **get**; **set**; }

**private** String? Message { **get**; **set**; }

**public** String Address

{

**get**

{

**return** $"{Index}, {City}, {Street}, {House}/{Building}, {Flat}";

}

}

*// Empty constructor*

**public** Mail() { }

*// Constructor with params*

**public** Mail(String? index, String? city, String? street, String? house, String? building, String? flat, String? message)

{

Index = index;

City = city;

Street = street;

House = house;

Building = building;

Flat = flat;

Message = message;

}

*// Copy constructor*

**public** Mail (Mail previousMail)

{

Index = previousMail.Index;

City = previousMail.City;

Street = previousMail.Street;

House = previousMail.House;

Building = previousMail.Building;

Flat = previousMail.Flat;

Message = previousMail.Message;

}

*// Clone method*

**public** Mail Clone()

{

**return** **new** Mail(Index, City, Street, House, Building, Flat, Message);

}

*// Method Valid()*

**public** bool Valid()

{

**return** Index != **null** && City != **null** && Street != **null** && House != **null** && Building != **null** && Flat != **null** && Message != **null**;

}

}

**internal** **class** **Program**

{

**public** **static** **void** Main(String[] args)

{

Mail \_1st = **new** (

args.Length >= 1 ? args[0] : **null**,

args.Length >= 2 ? args[1] : **null**,

args.Length >= 3 ? args[2] : **null**,

args.Length >= 4 ? args[3] : **null**,

args.Length >= 5 ? args[4] : **null**,

args.Length >= 6 ? args[5] : **null**,

args.Length >= 7 ? args[6] : **null**);

PrintMail(\_1st);

Mail \_2nd = **new** ("191023", "Saint-Petersburg", "Nevskiy Avenue", "77", "1", "21", "Hello!");

PrintMail(\_2nd);

Mail \_3rd = **new** (\_1st);

PrintMail(\_3th);

Mail \_4th = \_2nd.Clone();

PrintMail(\_4th);

}

**private** **static** **void** PrintMail(Mail mail)

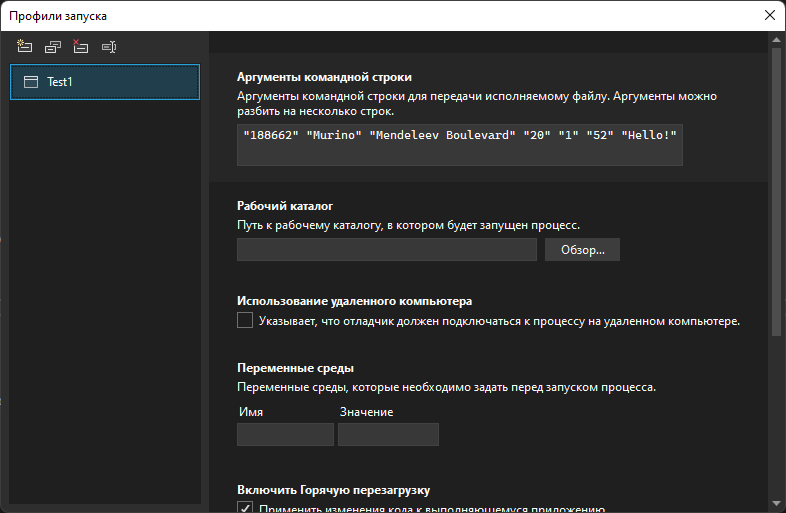
{

Console.WriteLine("Address: '" + mail.Address + "' is valid: " + mail.Valid());

}

}

Рисунок 1 - Аргументы командной строки



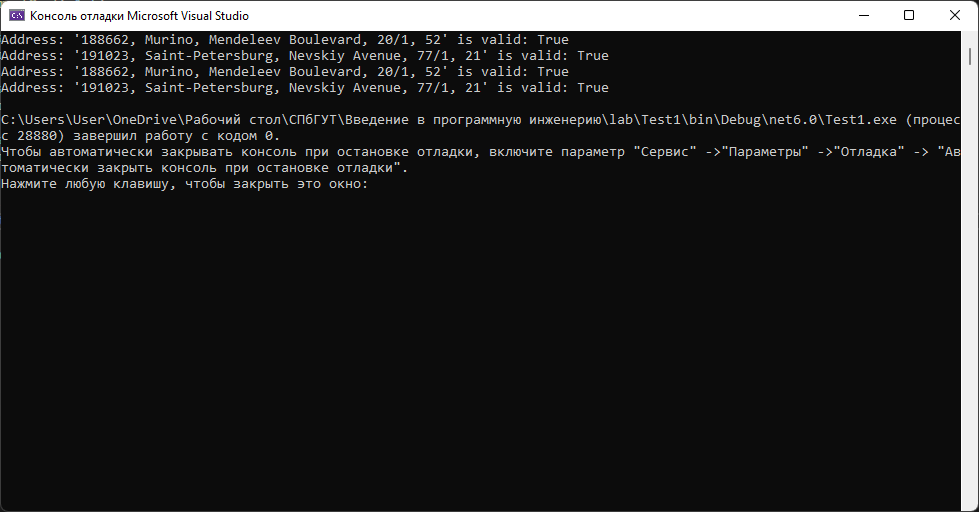


Рисунок 2 - Результат работы программы

**Эксперимент 2**

В данном проекте, помимо статического метода PrintMail, который выводит информацию об объекте, присутствует статический метод PrintMails, в теле которого объявляется цикл foreach и выводится вся информация об объектах коллекции mails.

В теле метода Main объявляется коллекция объектов класса Mail, затем с помощью цикла for создаются новые объекты для коллекции mails и одновременно описываются свойства новых объектов. Из коллекции с помощью метода RemoveAt() удаляются объекты с индексами 1 и 3. Удаление можно осуществить двумя способами:

1. Удалить объект с индексом 1, а затем, так как в коллекции произойдёт смещение объектов, удалить объект с индексом 2.
2. Сначала удалить объект с индексом 3, избегая нарушения порядка для первых элементов коллекции, потом – с индексом 1.

В конце тела метода Main вновь выводится информация обо всех элементах коллекции.

**internal** **class** **Program**

{

**public** **static** **void** Main(String[] args)

{

List<Mail> mails = **new** List<Mail>();

**for** (int i = 0; i < 4; i++)

{

mails.Add(**new** Mail("Индекс-" + i, "Город-" + i, "Улица-" + i, "Дом-" + i, "Корпус-" + i, "Квартира-" + i, "Сообщение-" + i));

}

PrintMails(mails);

Console.WriteLine("\n");

*// Remove 2nd and 4th objects*

*/\* mails.RemoveAt(1)*

*\* mails.RemoveAt(2) \*/*

mails.RemoveAt(3);

mails.RemoveAt(1);

PrintMails(mails);

}

**private** **static** **void** PrintMail(Mail mail)

{

Console.WriteLine("Address: '" + mail.Address + "' is valid: " + mail.Valid());

}

**private** **static** **void** PrintMails(List<Mail> mails)

{

**foreach** (Mail mail **in** mails)

{

PrintMail(mail);

}

}

}

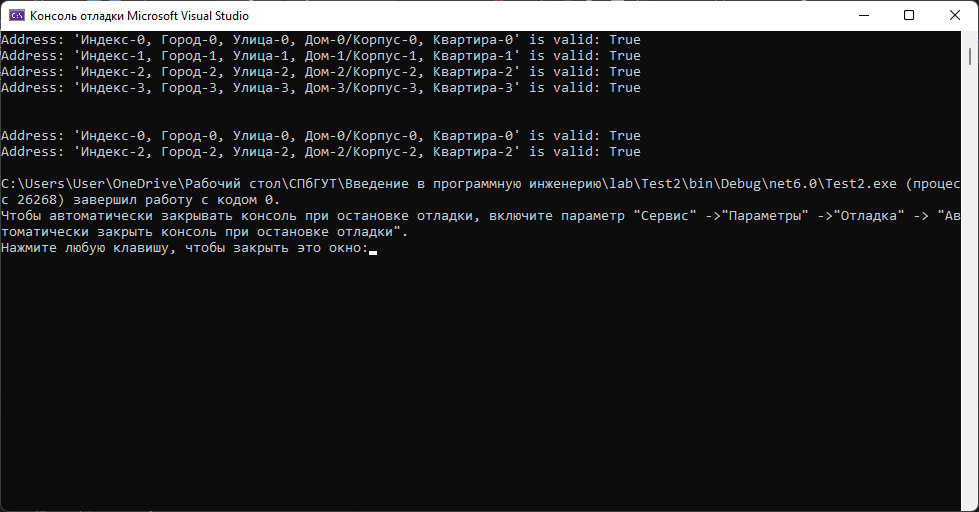
**Вывод**

Рисунок 3 - Результат работы программы

Язык программирования C# и интегрированная среда программирования Microsoft Visual Studio не трудны в освоении и предоставляют для начинающих удобный и негромоздкий функционал.

При выполнении поставленных задач цель была достигнута: приобретение навыков создания классов и объектов произошло без осложнений.

**Список используемых источников**

* «C#. Практическое руководство», Евдокимов П.В., 2022
* METANIT.COM, «Классы и объекты»: <https://metanit.com/sharp/tutorial/3.1.php>
* METANIT.COM, «Конструкторы, инициализаторы и деконструкторы»: <https://metanit.com/sharp/tutorial/3.35.php>
* learn.microsoft.com, «Свойства»: <https://learn.microsoft.com/ru-ru/dotnet/csharp/programming-guide/classes-and-structs/properties>