# Лабораторная работа № 12-13

**Тема:** Разработка приложений с использованием классов в С#.

**Цель работы:**

1. формирование навыков разработки программ на языке С# с использованием классов;
2. изучение способов описания и использования классов на языке С#.

. **Оборудование:**

1. ПК
2. Программное обеспечение: ОС Windows, среда Visual Studio.Net

**Контрольные вопросы:**

**1. Что такое класс?**

Класс в C# - это конструкция языка, которая представляет собой шаблон или описание объекта. Он определяет набор свойств, методов и событий, которые могут быть использованы для создания экземпляров (объектов) данного класса.

**2. Как описать класс на С# ?**

Для описания класса на C# нужно использовать ключевое слово class, за которым следует имя класса. Описание класса и его членов помещаются внутри фигурных скобок {} после объявления имени класса.

**3. Что такое спецификаторы и для чего они используются?**

Спецификаторы доступа (access modifiers) в C# используются для определения уровня доступа к классам, членам класса и другим элементам программы. Они определяют, кто и из каких частей вашего кода может обращаться к определенным элементам.

**4. Что относится к членам класса?**

Члены класса - это переменные, методы, свойства, константы и другие элементы, объявленные внутри класса. Они относятся к определенному классу и могут быть использованы для работы с объектами этого класса.

**5. Как задать переменные класса?**

Переменные класса объявляются внутри класса, но вне методов или конструкторов. Они являются членами класса и могут быть использованы внутри любого метода класса. Для объявления переменной класса нужно указать ее тип, имя и при желании начальное значение.

**6. Как задать методы класса**

Методы класса определяются внутри класса и выполняют определенные действия. Они могут принимать параметры и возвращать значения. Методы могут быть объявлены с различными спецификаторами доступа, их можно называть или без параметров, или с параметрами.

**7. Какие способы доступа к переменным класса используются в С#?**

В C# используются различные способы доступа к переменным класса:

- public - переменная доступна из любой точки программы.

- private - переменная доступна только изнутри самого класса.

- protected - переменная доступна только изнутри самого класса и его производных классов.

- internal - переменная доступна только в пределах текущей сборки.

- protected internal - переменная доступна внутри самого класса, его производных классов и в пределах текущей сборки.

**8. Для чего используют конструкцию this?**

Ключевое слово this в C# ссылается на текущий экземпляр объекта класса. Оно используется для разрешения конфликтов идентификаторов между параметрами методов и переменными класса, а также для доступа к другим членам класса.

**9. Что такое инкапсуляция?**

Инкапсуляция в C# - это концепция объектно-ориентированного программирования, которая позволяет объединять данные и методы в классе и скрывать их от общего доступа. Инкапсуляция обеспечивает защиту данных от неправильного использования и облегчает изменение внутренней реализации класса без влияния на другие части программы.

**10. Для чего используют в классах на С# свойства?**

Свойства в C# используются для определения геттеров и сеттеров для доступа к полям класса. Они обеспечивают контроль над доступом к данным и позволяют выполнить дополнительные действия во время получения или установки значения.

**11. Как задать свойства в программе с классами?**

Чтобы задать свойства в программе с классами на C#, нужно объявить свойство внутри класса, указав его тип, имя и определение геттера и/или сеттера. Геттер возвращает значение свойства, а сеттер устанавливает значение свойства.

**12. Для чего используют set и get в программах с классами на С#**

В программах с классами на C# ключевые слова set и get используются для определения геттеров и сеттеров свойств. set используется для установки значения свойства, а get используется для получения значения свойства. Они позволяют задавать изначальные правила доступа к свойствам и выполнение дополнительных действий при получении или установке значения свойства.

**Ход работы:**

1. Разработать алгоритм задачи и представить его в виде схемы программы
2. Используя средства среды Visual Studio.Net создать файл с программой и выполнить тестирование и отладку.
3. Результаты представить в виде отчета
4. Сделать вывод о проделанной работе

**Задание на лабораторную работу**

Разработать класс для объекта Train: Пункт назначения, Номер поезда, Время отправления, Число общих мест, Купейных, Плацкартных. Создать массив объектов. Вывести: а) список поездов, следующих до заданного пункта назначения; б) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа; в) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места.

**Листинг с исходным кодом**

using System;

using System.Net;

class Train

{

public string Destinat; // Пункт назначения

public int TrainNumber; // Номер поезда

public string DepartureTime; // Время отправления

public int TotalSeats; // Число общих мест

public int CoupeSeats; // Число купейных мест

public int ReservedSeats; // Число плацкартных мест

public void Print()

{

Console.WriteLine($"Пункт назначения:{Destinat} \t Номер поезда: {TrainNumber}\t Время отправления: {DepartureTime}\t Число общих мест: {TotalSeats}\t Число купейных мест: {CoupeSeats}\t Число плацкартных мест: {ReservedSeats}");

}

}

class Program

{

static void Main()

{

Console.WriteLine("Введите количество поездов: ");

int count = Convert.ToInt32(Console.ReadLine());

Train[] trains = new Train[count];

for (int i = 0; i < count; i++)

{

Console.WriteLine("Введите данные для поезда {0}: ", i + 1);

Console.Write("Пункт назначения: ");

string destinat = Convert.ToString( Console.ReadLine());

Console.Write("Номер поезда: ");

int trainNumber = Convert.ToInt32(Console.ReadLine());

Console.Write("Время отправления : ");

string departureTime = Console.ReadLine();

Console.Write("Число общих мест: ");

int totalSeats = Convert.ToInt32(Console.ReadLine());

Console.Write("Число купейных мест: ");

int coupeSeats = Convert.ToInt32(Console.ReadLine());

Console.Write("Число плацкартных мест: ");

int reservedSeats = Convert.ToInt32(Console.ReadLine());

trains[i] = new Train

{

Destinat = destinat,

TrainNumber = trainNumber,

DepartureTime = departureTime,

TotalSeats = totalSeats,

CoupeSeats = coupeSeats,

ReservedSeats = reservedSeats

};

}

Console.WriteLine("\nСписок поездов до заданного пункта назначения: ");

string destant = Console.ReadLine();

Console.WriteLine("а)");

for (int i = 0; i < trains.Length; i++)

{

if (trains[i].Destinat == destant) trains[i].Print();

}

Console.WriteLine("\nСписок поездов до заданного пункта назначения и отправляющихся после заданного часа: ");

string departTime = Console.ReadLine();

Console.WriteLine("б)");

for (int i = 0; i < trains.Length; i++)

{

if (trains[i].Destinat == destant && trains[i].DepartureTime == departTime) trains[i].Print();

}

Console.WriteLine("\nСписок поездов, отправляющихся до заданного пункта назначения и имеющих общие места: ");

int totSeats =Convert.ToInt32(Console.ReadLine());

Console.WriteLine("в)");

for (int i = 0; i < trains.Length; i++)

{

if (trains[i].Destinat == destant && trains[i].TotalSeats == totSeats) trains[i].Print();

}

Console.ReadKey();

}

}

**Результаты тестирования**

