# **Практическая работа № 1. Классы**

Объём учебного времени – 2ч

Методические рекомендации

1. Цель работы:Научиться работать со строками и символами.
2. Перечень необходимых средств обучения:
   1. Технические средства обучения:

Компьютер Core i3 3.0, 2 Gb оперативной памяти, винчестер 250 Gb, DVD

* 1. Программное обеспечение:
* Программа ОС Windows XP/7;
* Антивирусные программы: KasperskyAntiVirus;
* Браузер InternetExplorer;
* Интегрированная среда программирования MSVisualStudio 2017.

1. Основные теоретические положения:

Класс – разновидность абстрактного типа данных в ООП, характеризуемый способом своего построения. Класс представляет собой шаблон для создания объектов.

Объекты являются экземплярами классов.

Классы, как и получаемые по их описаниям объекты, состоят из объединения членов (инкапсуляция) – данных-полей и кода-методов.

Методы объектов близкой спецификации могут иметь одинаковые имена и назначение, но выполнять действия, характерные для конкретного объекта (полиморфизм).

При необходимости получения новых возможностей можно добавить дочернему классу недостающие члены, позаимствовав уже существующие у родительского класса (наследование).

Пример использования класса:

Using System;

namespace HelloApp

{

    Class Person

    {

        public string name; // имя

        public int age;     // возраст

        public void GetInfo()

        {

            Console.WriteLine($"Имя: {name}  Возраст: {age}");

        }

    }

    Class Program

    {

        Static void Main(string[] args)

        {

            Person tom;

            Console.ReadKey();

}

}

}

### Классы: основные понятия

Класс является типом данных, определяемым пользователем. Он должен представлять собой одну логическую сущность, например, являться моделью реального объекта или процесса. Элементами класса являются данные и функции, предназначенные для их обработки.

### Описание класса имеет вид:

[атрибуты] [спецификаторы] class <имя\_класса> [: предки]

{ тело\_класса }

Обязательными являются только ключевое слово class, а также имя и тело класса. Имя класса задается программистом по общим правилам С#. Тело класса – это список описаний его элементов, заключенный в фигурные скобки. Спецификаторы определяют свойства класса, а также доступность класса для других элементов программы. Класс можно описывать непосредственно внутри пространства имен или внутри другого класса. Для классов, которые описываются в пространстве имен непосредственно, допускаются только два спецификатора: public и internal. По умолчанию подразумевается спецификатор internal.

Класс является обобщенным понятием, определяющим характеристики и поведение некоторого множества конкретных объектов, и используется для их создания. Объекты также называются экземплярами класса. Объекты создаются явным или неявным образом, то есть либо программистом, либо системой.

Класс относится к ссылочным типам данных, память под которые выделяется в хипе. Для каждого объекта при его создании в памяти выделяется отдельная область, в которой хранятся его данные.

В классе могут присутствовать элементы, которые существуют в единственном экземпляре для всех объектов класса – статические элементы.

Функциональные элементы класса не тиражируются, то есть всегда хранятся в единственном экземпляре. Для работы со статическими данными класса используются статические методы, для работы с данными экземпляра – методы экземпляра, или просто методы.

### Данные: поля и константы

Данные, содержащиеся в классе, могут быть переменными или константами. Переменные, описанные в классе, называются полями класса.

Синтаксис описания элемента данных: [атрибуты][спецификаторы][const] <тип> <имя> [= <начальное значение>]

По умолчанию элементы класса считаются закрытыми

(private). Все методы класса имеют непосредственный доступ к его закрытым полям.

Все поля сначала автоматически инициализируются нулем соответствующего типа. После этого полю присваивается значение, заданное при его явной инициализации. Задание начальных значений для статических полей выполняется при инициализации класса, а обычных – при создании экземпляра.

Поля, описанные со спецификатором static, а также константы существуют в единственном экземпляре для всех объектов класса, поэтому к ним обращаются через имя класса. Обращение к полю класса выполняется с помощью операции доступа - точки:

для обычных полей: <имя экземпляра>.<имя поля>

для статических: <имя класса>.<имя поля>

Пример 1. Класс Demo, содержащий поля и константу.

using System;

namespace ConsoleApplicationl

{class Demo

{ public int a = 1; //поле данных

public const double с = 1.66; //константа

public static string s = "Demo";//статич. поле класса

double y; //закрытое поле данных}

class Classl

{static void Main()

{Demo x = new Demo();//создание экземпляра класса Demo Console.WrteLine(x.a); //обращение к полю класса Console.WriteLine(Demo.с); // обращение к константе Console.WriteLine(Demo.s); //обращение к статич. полю }}} Поле y вывести на экран аналогичным образом нельзя – оно является закрытым, т.е. недоступно из класса Classl. Поскольку значение этому полю явным образом не присвоено, среда присваивает ему значение ноль.

### Методы

Методы класса имеют непосредственный доступ к его закрытым полям. Метод, описанный со спецификатором static, должен обращаться только к статическим полям класса. Статический метод вызывается через имя класса, а обычный – через имя экземпляра.

### Конструкторы

Конструктор предназначен для инициализации объекта. Он вызывается автоматически при создании объекта класса с помощью операции new. Формат записи конструктора такой:

[спецификатор] <имя\_класса>()

{тело конструктора}

Обычно в качестве элемента спецификатор используется модификатор доступа public, поскольку конструкторы, как правило, вызываются вне их класса.

Класс может иметь несколько конструкторов с разными параметрами для разных видов инициализации. Конструктор, вызываемый без параметров, называется конструктором по умолчанию. Если в классе не указан ни один конструктор или какие-то поля не были инициализированы, полям значимых типов присваивается нуль соответствующего типа, полям ссылочных типов – значение null.

Создание объекта выполняется операцией

<имя\_переменной\_типа\_класса> = new <имя\_класса>();

Имя класса вместе со следующей за ним парой круглых скобок – это не что иное, как конструктор реализуемого класса. Если в классе конструктор не определен явным образом, оператор new будет использовать конструктор по умолчанию, который предоставляется средствами языка С#.

Пример 2. В программе создаются два объекта с различными значениями полей с

1. Практические указания:
   1. Создать класс для реализации методов решения квадратного уравнения ax2+bx+c=0. Уравнение задается набором коэффициентов (от 1 до 3). Если при создании указывается иное количество коэффициентов, то квадратное уравнение определить нельзя, поэтому выдается предупреждение об ошибке. В классе должны быть предусмотрены средства для решения уравнений, в которых a=0, b=0 или c=0. Тогда уравнение может стать линейным (0\*x2+5x+2=0), обратиться в тождество (0 = 0) или стать неразрешимым (6 = 0).
   2. Создайте консольное приложение.
   3. Объявите класс Equation.
   4. Определите поля класса:
   * константу для обозначения бесконечного количества корней



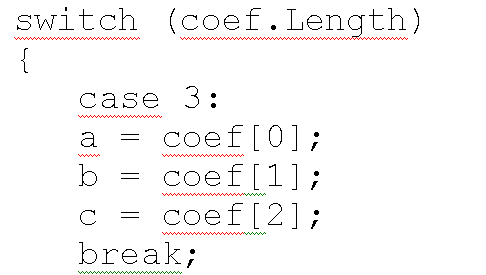
* + вещественные переменные для хранения коэффициентов уравнения a,b,c;
  + целочисленное поле для хранения количества корней уравнения count=-1

(-1 означает, что вычисление корней уравнения пока не осуществлялось);

* + вещественные переменные для хранения корней уравнения x1,x2;
  1. Определитеконструктор класса. Для создания объекта класса Equation требуется задать его коэффициенты. Поскольку количество коэффициентов может быть переменным (от 1 до 3), конструктор можно сделать методом с переменным числом параметров (определенным с ключевым словом params).



* + в зависимости от количества параметров (длины массива) получаем различные виды уравнений, для этого используем оператор switch-case:

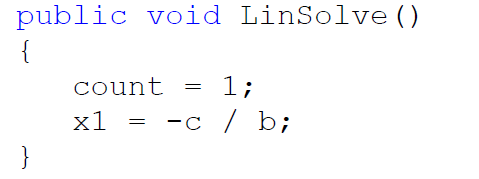


* + продумайте запись выбора для двух и одного коэффициентов;
  + по умолчанию, средствами оператора throw сгенерируйте исключение при некорректном наборе коэффициентов.
  1. Опишите метод Solve(). Он предназначен для определения порядка уравнения (квадратное – уравнение 2-ого порядка, линейное – уравнение 1-ого порядка, тождество или неразрешимое – уравнение 0-ого порядка).

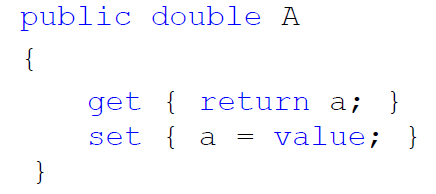


* + метод содержит вложенный оператор ветвления и последовательно проверяет:

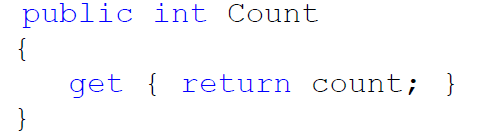
1. если отсутствует первый коэффициент , то выполняем проверку:
2. отсутствует ли второй коэффициент, то проверяем:
3. отсутствует ли третий, тогдаколичество корней равно бесконечному количеству, иначе: количество корней равно 0;
4. если уравнение не линейное, то вызываем функцию для решения уравнений 1-ого порядка, назовем ее LinSolve();
5. если уравнение не квадратное, то вызываем функцию для решения уравнений 2-ого порядка, назовем ее QSolve();
   1. Опишите метод решения квадратного уравнения QSolve(); Для этого:
   * Опишите вещественную переменную disc, для вычисления дискриминанта и вычислите его;
   * Выполните проверку: если дискримининт меньше 0, то вещественных корней нет (значение переменной count=0);
   * Иначе, если дискриминант равен 0, то уравнение имеет один корень. Рассчитайте этот корень, запишите его значение в переменную x1;
   * Иначе, уравнение имеет два корня. Рассчитайте эти корни, запишите их значения в переменные x1 и x2;
   1. Опишите метод решения линейного уравнения LinSolve();



* 1. Определите свойства для доступа к коэффициентам уравнения и получения количества его корней. Согласно принципу инкапсуляции данные класса являются закрытыми (private). К ним можно обращаться только из методов самого класса. В случае, когда требуется изменить или получить значения тех или иных данных класса следует определять специальные методы доступа к данным (обычно называются get- и set-методы). Эти методы оформляются в виде свойств класса.
  + Определите свойства для обеспечения доступа к коэффициентам уравнения:



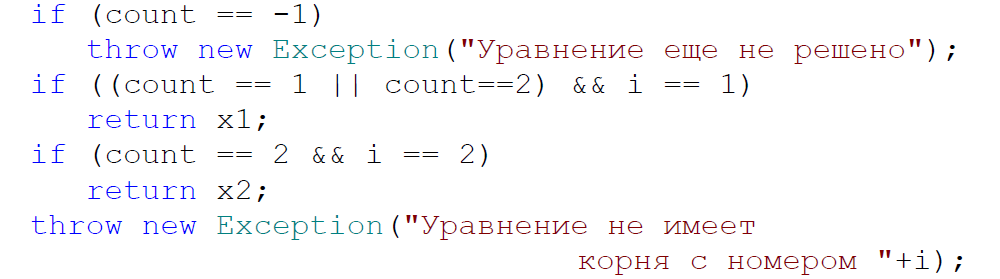
* + По аналогии определите свойства для обеспечения доступа к коэффициентам b и c;
  1. Создайте также свойство Сount, для получения количества корней:



* 1. Определите индексатор для получения корней уравнения по номеру.Отличием индексаторов от свойств является то, что индексатор осуществляет доступ к отдельному элементу из упорядоченного набора значений. В качестве индекса здесь используется целое число (1 или 2), соответствующее номеру корня.



* + Для получения корня по номеру используйте метод get;
  + Запишите и разберите следующий код:



* 1. Вывести корни уравнения, для этого опишите метод PrintSolution(). В методе используйте оператор switch – case. В качестве выражения – селектора используйте переменную count. Выводимое сообщение определяется значением количества корней уравнения: 0,1 или 2, также предусмотрите сообщение для любого значения корня и нерешенного уравнения.
  2. В главном модуле используйте оператор try-catch, поскольку в конструкторе может быть сгенерировано исключение. В блоке try:
  + создайтеобъект-уравнение,
  + примените метод для решения уравнения;
  + выведите результаты решения;
  + в блоке catch – выведите сообщение об ошибке.
  1. Индивидуальный вариант

Вариант 1

Предметная область: ATC. В классе хранить информацию об адресе АТС, числе абонентов, абонентской плате (для всех абонентов одна). Реализовать метод для подсчета абонентской платы всех клиентов.

Вариант 2

Предметная область: Вокзал. В классе хранить информацию о наименовании станции, стоимости билета (стоимость одинакова для всех направлений), числе мест, числе проданных билетов. Реализовать метод для подсчета общей стоимости всех непроданных билетов.

Вариант 3

Предметная область: ЖЭС. В классе хранить информацию о районе, к которому принадлежит ЖЭС, номере ЖЭС, числе жильцов, оплате за месяц (для всех жильцов одна), числе оплативших. Реализовать метод для подсчета общей задолженности жильцов.

Вариант 4

Предметная область: Аэропорт. В классе хранить информацию о названии аэропорта, стоимости билета (стоимость одинаковая), общем числе мест во всех самолетах, числе проданных биле-тов. Реализовать метод для подсчета общей стоимости всех проданных билетов.

Вариант 5

Предметная область: Банк. В классе хранить информацию о наименовании банка, числе вкладов, размере вклада (все вклады одинаковые), размере процентной ставки. Реализовать метод для подсчета общей выплаты по процентам.

Вариант 6

Предметная область: Отдел кадров. В классе хранить информацию о наименовании предприятия, числе работников, норме выработки часов в месяц (одна для всех работников), оплате за час, подоходном налоге. Реализовать метод для подсчета общей выплаты по подоходному налогу.

Вариант 7

Предметная область: Фирма грузоперевозок. В классе хранить информацию об оплате за перевозку одной тонны грузов (не зависит от направления), о массе перевезенных грузов, наименовании фирмы. Реализовать метод для подсчета общей выручки фирмы.

Вариант 8

Предметная область: Гостиница. В классе хранить информацию о названии гостиницы, числе заселенных мест, общем числе мест, оплате за день проживания (для всех жильцов одинаковая стоимость). Реализовать метод для подсчета общей выручки гостиницы.

Вариант 9

Предметная область: Интернет-оператор. В классе хранить информацию о стоимости тарифа (одна для всех пользователей), наименовании оператора, числе абонентов. Реализовать метод для подсчета общей выручки.

Вариант 10

Предметная область: Интернет-магазин по продаже телевизоров. В классе хранить информацию о стоимости телевизора (одна для всех моделей), наименовании магазина, числе покупок. Реализовать метод для подсчета общей выручки.

Вариант 11

Предметная область: Завод. У работника завода хранить фамилию, год рождения, размер почасовой оплаты и количество отработанных часов. В классе реализовать метод для подсчета заработной платы работника, исходя из величины почасовой оплаты и отработанных часов.

Вариант 12

Предметная область: Банковский счет. В классе хранить информацию о номере банковского счета, фамилию, имя, отчество, владельца счета, сумму вклада и дату внесения вклада. В классе реализовать метод для подсчета процентов по вкладу, считая, что клиент получает 6,8 процентов годовых от внесенной суммы.

1. Контрольные вопросы
2. Чтотакоекласс?
3. Чтотакоеобъект?
4. КакимисвойствамиобладаетООП?
5. Длячегоиспользуютсяконструкторы?
6. Какоймодификаторпозволяетобращатьсякчленамизвнекласса?
7. Список рекомендуемой литературы:

**Основная литература:**

1. Немцова Т.И. Программирование на языке высокого уровня. Программирование на языке ObjectPascal: учеб. пособие /Т.И. Немцова, С.Ю. Голова, И.В. Абрамова. – М.: ФОРУМ: ИНФРА – М, 2012. – 496 с.
2. Овечкин Г.В. Компьютерное моделирование: учебник для студ. учреждений сред. проф. образования / Г.В. Овечкин.- М.: Издательский центр «Академия», 2015.-224с.
3. Немцова, Т. И.  Программирование на языке высокого уровня. Программирование на языке С++ : учеб. пособие / Т. И. Немцова, С. Ю. Голова, А. И. Терентьев. - М. : ФОРУМ : ИНФРА-М, 2012. - 512 с. : ил. + CD.
4. Гуриков, С. Р. Введение в программирование на языке Visual C# : учеб. пособие / С. Р. Гуриков. - М. : ФОРУМ : ИНФРА-М, 2013. - 448 с.

**Дополнительная литература:**

1. Рао Сиддхартха Освой самостоятельно С++ за 21 день, 7 изд.: Пер с англ.-М.: ООО «И.Д.Вильямс», 2013 – 688 с.: ил. – Парал.тит.англ.
2. Голицына О.Л. Программное обеспечение: учеб. пособие для среднего профессионального образования-М.:ФОРУМ;ИНФРА-М,2006.-432 с.
3. Виллемер А. Программирование на С++/А. Виллемер;[пер. с нем. М.А.Райтман].-М.:Эксмо,2013.-528с.+CD.-(Мировой компьютерный бестселлер).
4. Культин Н.Б. MicrosoftVisualC++ в задачах и примерах.- СПб.: БХВ-Петербург,2010.-272 с.:ил.+CD-ROM.
5. Партыка Т.Л. Операционные системы, среды и оболочки: учеб. пособие для студ. учреждений сред. проф. Образования/ Т.Л. Партыка, И.И. Попов. – 3-е изд., перераб. и доп. – М.: ФОРУМ, 2010. – 543 с.
6. Окулов С.М. Основы программирования: учебное пособие.- М.:БИНОМ. Лаборатория знаний, 2010.- 440 с.
7. Голицына О.Л. Основы алгоритмизации и программирования: учеб. пособие для сред. проф. образования.- М: ФОРУМ; ИНФРА-М,2005.-432 с.
8. Рихтер Дж. Программирование приложений для MicrosoftWindows /Пер. с англ. – M.: MicrosoftPress, 2003. – C.48-313.