**ПРАКТИЧЕСКАЯ РАБОТА №5**

**Классы в С#**

****

**Цель работы**: научиться разрабатывать классы, их основныекомпоненты: поля, методы, конструкторы, создавать объекты класса, обращаться к открытым полям и методам класса.

Для защиты практической работы необходимо изучить теоретическую часть, выполнить практическое задание, ответить на вопросы.

|  |  |  |
| --- | --- | --- |
| №  п/п | Ф.И.О. | НОМЕР ВАРИАНТА/ЗАДАНИЯ |
|  | Абабий Илья | ПРИМЕР ИЗ ТЕОРЕТИЧЕСКОЙ ЧАСТЬ С ДОБАВЛЕНИЕМ 1 ПОЛЯ |
|  | Ботнарь Артем | 1 (с добавлением еще 2 полей) |
|  | Войнован Леонид | 2 |
|  | Гулин Максим | 3 (с добавлением еще 2 полей) |
|  | Дубчак Егор | 4(с добавлением еще 2 полей) |
|  | Дяченко Ольга | ПРИМЕР ИЗ ТЕОРЕТИЧЕСКОЙ ЧАСТЬ С ДОБАВЛЕНИЕМ 1 ПОЛЯ |
|  | Игнатенко Максим | ПРИМЕР ИЗ ТЕОРЕТИЧЕСКОЙ ЧАСТЬ С ДОБАВЛЕНИЕМ 1 ПОЛЯ |
|  | Крижановская Арина | 5 |
|  | Лебедева Серафима | 6 (с добавлением еще 3 полей) |
|  | Лунгу Никита | 1 |
|  | Марков Кирилл | 8 (с добавлением еще 2 полей) |
|  | Мустяца Андрей | 1 (с добавлением еще 2 полей) |
|  | Мутаф Артем | 10 (с добавлением еще 3 полей) |
|  | Накул Вячеслав | 11 (с добавлением еще 3 полей) |
|  | Овчарук Дарья | 12(с добавлением еще 3 полей) |
|  | Полозюк Даниил | 13 |
|  | Почтарь Александр | 14 |
|  | Ратушинский Михаил | 15(с добавлением еще 2 полей) |
|  | Ривилис Евгений | 9 (с добавлением еще 2 полей) |
|  | Салтыков Антон | ПРИМЕР ИЗ ТЕОРЕТИЧЕСКОЙ ЧАСТЬ С ДОБАВЛЕНИЕМ 1 ПОЛЯ |
|  | Сиглов Артем | 7 |
|  | Ткачук Анастасия | 5 (с добавлением еще 3 полей) |
|  | Цвигуненко Юлия | 10 (с добавлением еще 3 полей) |
|  | Черных Артем | 3 |
|  | Шурдук Виктория | 5 (с добавлением еще 3 полей) |
|  | Щуровская Александра | 4 (с добавлением еще 3 полей) |
|  | Янишевский Владимир | 11 (с добавлением еще 3 полей) |

**Теоретические сведения**

Основным понятием объектно-ориентированного подхода является класс. Любая программная система, разработанная на основе объектно-

ориентированного подхода, содержит ансамбль классов, взаимодействующих между собой.

* объектно-ориентированном программировании класс играет две роли: модуля и типа данных.

Первая роль. Класс - это модуль, архитектурная единица построения программной системы.

Вторая роль. Класс - это тип данных, задающий реализацию некоторой сущности рассматриваемой предметной области. Состав класса, его размер определяется информационной моделью рассматриваемой сущности реального мира.

Объектно-ориентированная разработка программной системы основана на стиле, называемом проектированием от данных. Проектирование системы сводится к поиску абстракций данных, подходящих для конкретной задачи. Каждая из таких абстракций реализуется в виде класса, которые и становятся модулями - архитектурными единицами построения нашей системы. В основе класса лежит абстрактный тип данных.

* хорошо спроектированной программе каждый класс играет обе роли, то есть каждый класс программы имеет вполне определенную смысловую нагрузку.

Синтаксис описания класса:

**[атрибуты][модификаторы]*class* имя\_класса[:список\_родителей] {тело\_класса}**

Значениями модификатора в объявлении класса могут быть: ***new*,** ***abstract, sealed***, которые связаны с наследованием, и модификаторы доступа,причем два из них - ***private*** и ***protected*** - могут быть заданы только для вложенных классов. Обычно класс имеет модификатор доступа ***public***, задающийся по умолчанию.

Пример объявления класса: *public class Rational* {тело\_класса}тело\_класса} В теле класса могут быть объявлены:

константы;

поля;

конструкторы и деструкторы;

методы;

события;

делегаты;

вложенные классы (структуры, интерфейсы, перечисления). Классы могут быть вложенными, т.е. описание одного класса может

содержать описание другого класса. Так поступают, если вложенный класс носит вспомогательный характер и разрабатывается в интересах одного конкретного класса, при этом разработчик считает, что вложенный класс никому не понадобится, кроме класса, в который он вложен. Вложенные классы обычно имеют модификатор доступа, отличный от *public*.

Основу любого класса составляют его конструкторы, поля и методы.

Рассмотрим эти компоненты класса более подробно:

**Поле**. Так называется член-переменная, содержащий некотороезначение. Их иногда называют данными объекта. К полю можно применять несколько модификаторов в зависимости от того, как вы собираетесь его использовать, по умолчанию модификатор доступа поля - *private*.

**Метод**. Это некоторая функция, операторы которой,воздействуют на поля класса.

**Свойства**. Их называют «разумными» полями (*smart fields*),поскольку они являются методами, которые клиенты класса воспринимают как поля. Они обязательно включают в себя хотя бы один из аксессоров – *set*, *get.* Свойства позволяют клиенту класса произвести запись значения в полекласса или чтение содержимого поля после некоторой обработки, которое обеспечивает выполнение ряда ограничений, накладываемых на содержимое поля.

**Константы**. Это поле, значение которого изменить нельзя.

**Индексаторы**. Это «разумные» массивы, они позволяютиндексировать объекты методами-аксессорами *get* и *set*. С помощью индексатора легко проиндексировать объект для установки или получения значений.

**События**. Событие вызывает исполнение некоторой функции –обработчика события.

Поля класса синтаксически являются обычными переменными (объектами) языка. Их описание удовлетворяет обычным правилам объявления переменных. Поля содержат значения характеристик объекта класса.

При создании каждого нового объекта класса в динамической памяти или в стеке, этот объект представляет собой набор полей класса. Два объекта одного класса имеют один и тот же набор полей, но разнятся значениями, хранимыми в этих полях. Например, все объекты класса *Pers* могут иметь поле *Vozr*, характеризующее возраст человека, но если мы создадим три объекта данного класса, характеризующие трех различных людей, то первый человек может иметь возраст 2 года, другой – 22 года, а третий – 42 года, что и будет отображено в полях соответствующих объектов класса *Pers*.

Каждое поле имеет модификатор доступа, принимающий одно из четырех значений: *public, private, protected, internal*. Описание модификаторов приведено в таблице 1.1.

|  |  |
| --- | --- |
|  | Таблица 1.1. - Модификаторы доступа |
|  |  |
| **Модификато** | **Описание** |
| **р доступа** |
|  |
| *Public* | Член доступен вне определения класса и иерархии производных классов. |
| *Protected* | Член невидим за пределами класса, к нему могут обращаться только |
|  | производные классы. |
| *Private* | Член недоступен за пределами области видимости определяющего его |
|  | класса. Поэтому доступа к этим членам нет даже у производных классов. |
| *Internal* | Член видим только в пределах текущей единицы компиляции. |
|  | Модификатор доступа *internal* в плане ограничения доступа является |
|  | гибридом *public* и *protected*, зависимым от местоположения кода. |

Значением модификатора доступа по умолчанию является *private*. Независимо от значения модификатора доступа, все поля и методы

класса доступны для всех методов класса. Они являются для методов класса источником информации, с которой работают все методы, извлекая из полей нужные им данные и изменяя их значения в ходе работы.

Если поля доступны только для методов класса, то они имеют значение модификатора доступа *private*, который можно опускать. Такие поля считаются закрытыми.

Если некоторые поля класса *A* должны быть доступны для методов класса B, являющегося потомком класса *A*, то эти поля следует снабдить атрибутом *protected*. Такие поля называются защищенными.

Если некоторые поля должны быть доступны для методов классов *Z1*,*Z2* и так далее, дружественных по отношению к классу *X*, то эти поля следуетснабдить атрибутом *internal*, а все дружественные классы X поместить в один проект (*assembly*). Такие поля называются дружественными.

Если некоторые поля должны быть доступны для методов любого класса *V*, которому доступен объект класса *W*, то эти поля следует снабдить атрибутом *public*. Такие поля называются открытыми.

Методы класса являются обычными процедурами и функциями языка. Их описание удовлетворяет правилам объявления процедур и функций. Методы содержат описания операций, доступных над объектами класса. Все объекты одного класса имеют один и тот же набор методов.

Каждый метод имеет модификатор доступа, принимающий одно из четырех значений: *public, private, protected, internal*. Атрибутом доступа по умолчанию является атрибут *private*. Независимо от значения атрибута доступа, все методы доступны для вызова при выполнении метода класса. Если методы имеют атрибут доступа *private*, возможно, опущенный, то тогда они доступны только для вызова внутри методов самого класса. Такие методы считаются закрытыми.

Класс, как правило, имеет открытые методы, задающие интерфейс класса, и закрытые методы. **Интерфейс** - это методы, через которые класс общается с окружающей средой, с их помощью организуется ввод данных, предоставление клиентам класса, результатов обработки данных и т.п.

Закрытые методы представляют собой ту часть класса, которая напрямую недоступна клиентам класса.

Если ряд методов класса AX должны быть доступны для вызовов в методах класса *BY*, являющегося потомком класса *AX*, то такие методы должны иметь значение модификатора *protected*. Если некоторые методы должны быть доступны только для методов классов *BY1, BY2* и так далее, дружественных по отношению к классу *AX*, то такие методы должны иметь значение модификатора доступа *internal*, а все дружественные классы *BY* поместить в один проект.

Если ряд методов должны быть доступны для методов любого класса *BY*, которому доступен сам класс *AX*, то такие методы должны иметьзначение модификатора доступа *public*.

Рассмотрим пример класса

*//Описание класса student*

*class student*

*{*

* *поля класса, все поля – закрытые, доступ к полям – через методы класса int oc1,oc2,oc3,oc4;*

*string fio, predmet1, predmet2, predmet3, predmet4;*

* *пустой конструктор класса*

*public student(){}*

*// конструктор класса*

*public student(string fio, string predmet1, string predmet2, string predmet3, string predmet4)*

*{*

*this.fio=fio;*

*this.predmet1= predmet1;*

*this.predmet2= predmet2;*

*this.predmet3= predmet3;*

*this.predmet4= predmet4;*

*}*

* *открытые методы класса public void pr1(int f)*

*{ this.oc1=f; }*

*public void pr2(int f)*

*{ this.oc2=f; } public void pr3(int f)*

*{ this.oc3=f; } public void pr4(int f)*

*{ this.oc4=f; } public void display()* {тело\_класса}

* *на экран выдается содержимое закрытых полей классса Console.WriteLine("студент {0}, оценки по {1}={2} по {3}={4} по {5}={6} по {7}={8}",*

*fio, predmet1, oc1, predmet2, oc2, predmet3, oc3, predmet4, oc4); Console.WriteLine("студент {0}, средний балл = {1}", fio, sredn\_ball()); Console.ReadLine();*

*}*

* *закрытый метод класса int sredn\_ball()*

*{ return (oc1+oc2+oc3+oc4)/4; } }// конец описания класса*

*static void Main(string[] args)*

*{*

*// создание объекта f класса student*

*student f=new student("Сидоров", "математика", "информатика",*

*"программирование", "история");*

*// обращение к открытым методам класса – ввод информации об оценках студента*

*f.pr1(2);*

*f.pr2(3);*

*f.pr3(3);*

*f.pr4(4);*

* *обращение к открытому методы класса – вывод результатов сессии f.display();*

*}*

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

1. Разработать класс, который ведет учет результатов сессии для конкретного студента и рекомендует: а) отчислить, б) сформировать график сдачи задолженностей, в) начислить стипендию, г) объявить благодарность, д) вынести выговор.
2. Разработать класс, который организует учет перемещения отдельной книги в библиотеке: постановка книги на учет, регистрация выдачи книги, регистрация возврата книги, списание книги, изменение стоимости книги.
3. Разработать класс, который организует учет студентов в общежитии: заселение студента, регистрация оплаты за проживание в течение года, переход в течение года из одной комнаты в другую, выселение студента, учет возможных взысканий/поощрений студента в течение года.
4. Из пункта А в пункт В вышел пешеход со скоростью *v*. Навстречу ему с опозданием на *t*1 часов вышел второй пешеход. Расстояние между пунктами А и В – *L* км. На каком расстоянии от пункта А, и в какое время они встретятся если первый пешеход шел на *х* км быстрее вто-рого. Решить задачу с использованием класса «Пешеход».
5. Река течет со скоростью *v*1 . По течению реки со скоростью *v* плывет корабль, который потратил на свой путь *t* часов. Узнать, сколько време-ни ему потребуется на обратный путь. Решить задачу с использованием класса «Корабль».
6. В бассейн объемом *V* поступает вода из одной трубы в количестве *V*3 куб м/час, а выливается из двух труб в количестве *V*1, *V*2 куб м/час. За сколько времени бассейн будет наполнен или опустеет? Решить задачу с использованием класса «Труба».
7. Из пункта *А* выехал велосипедист со скоростью *v*1. Одновременно вслед за ним из пункта *B*, отстоящего от *А* на *L* км выехал мотоциклист со скоростью *v*2. На каком расстоянии от пункта *А* мотоциклист догонит велосипедиста? Решить задачу с использованием класса «Задача».
8. Скашивая ежедневно по 60 га вместо 50, бригада сумела скосить луг на один день быстрее, чем планировалось. Какова площадь луга? Решить задачу с использованием класса «Бригада».
9. Разработать класс, который ведет учет результатов, полученных абитуриентом при поступлении в вуз. По итогам сессии для данного абитуриента формируется рекомендация: а) зачислить на бюджет, б) зачислить на договор, в) не зачислять.

Рассмотреть два случая:

- абитуриент подает документы на одну специальность;

- абитуриент подает документы сразу на 3 специальности.

10.Разработать класс, который бы позволял решать квадратные уравнения.

11.Дано алгебраическое выражение в виде частного двух полиномов. Порядок каждого полинома – не более 5. Каждый полином содержит ряд постоянных коэффициентов и переменную *x*. Разработать класс, позволяющий вычислить значение этого выражения для заданной величины *x*. Если знаменатель равен нулю, то возвращается значение 0 выдается сообщение об ошибке. Рекомендуется в классе создать не менее 2-х конструкторов.

12.Разработать класс, позволяющий выполнять арифметические операции (+ - \* / ) над комплексными числами.

13.Разработать класс, позволяющий вычислять стоимость коммунальных услуг для одной квартиры (квартплата, плата за отопление, плата за электричество, плата за газ). С помощью программы выдать на экран корешки на оплату коммунальных платежей для жителей 10-квартирного дома за январь, февраль, март. Учесть возможное наличие льгот.

14.Разработать класс, позволяющий начислить зарплату служащему. Форма оплаты – окладная. Необходимо учесть количество дней, отработанных в текущем месяце. Возможна премия (задается в процентах) и выплата из центрального фонда. Используя данный класс, произвести начисление зарплаты членам 12-го отдела (всего 5 человек) за март, апрель, май (сформировать расчетные листки).

15.Разработать класс, позволяющий начислить зарплату рабочему. Форма оплаты – сдельная. Рабочий изготавливает детали 5 видов. За каждую изготовленную деталь, каждого вида он получает некую сумму. Возможна премия (задается в процентах) и выплата из центрального фонда. Если сумма зарплаты меньше заданного числа, то выплачиваются «простойные», т.е. величина з/пл искусственно увеличивается до достижения заранее заданного минимума. Используя данный класс, произвести начисление зарплаты членам 7-й бригады (всего 5 человек) за март, апрель, май (сформировать расчетные листки).

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Что такое класс?
2. Что может входить в состав класса?
3. Чем отличается класс от структуры?
4. Модификаторы класса.
5. Модификаторы доступа к полям класса.
6. Модификаторы доступа к методам класса.
7. Что такое экземпляр класса? Как он создается?
8. Для чего в классе определяется конструктор?
9. Сколько может быть конструкторов в классе?
10. Когда вызывается конструктор?
11. Для чего в классе определяется деструктор?
12. Когда вызывается деструктор?
13. Как можно обратиться к полям и методам класса?