

Содержание

Введение	21
ЧАСТЬ I. НАЧНЕМ, ПОЖАЛУЙ...	29
Глава 1. Создание вашего первого консольного приложения на C#	31
Введение в машинные языки, C# и платформу .NET	31
Что такое программа	31
Что такое C#	32
Что такое .NET	33
Что такое Visual Studio 2008 и Visual C#	34
Создание первого консольного приложения	34
Создание исходной программы	34
Пробная поездка	37
Создание реального консольного приложения	37
Изучение консольного приложения	39
Схема программы	39
Комментарии	39
Тело программы	40
Использование Toolbox	41
Сохранение кода в Toolbox	41
Использование кода из Toolbox	42
ЧАСТЬ II. ОСНОВЫ ПРОГРАММИРОВАНИЯ В C#	43
Глава 2. Объявление переменных-значений	45
Объявление переменной	45
Что такое int	46
Правила объявления переменных	47
Вариации на тему int	47
Новый тип BigInteger	48
Представление дробных чисел	49
Работа с числами с плавающей точкой	49
Объявление переменной с плавающей точкой	50
Более точное преобразование температур	51
Ограничения переменных с плавающей точкой	51
Десятичные числа — комбинация целых и чисел с плавающей точкой	52
Объявление переменных типа decimal	52
Сравнение десятичных и целых чисел, а также чисел с плавающей точкой	53
Логичен ли логический тип	53
Символьные типы	54
Тип char	54
Специальные символы	54
Тип string	54

Что такое тип-значение	55
Сравнение <code>string</code> и <code>char</code>	56
Високосный ли этот год?	57
Объявление числовых констант	58
Преобразование типов	59
Новая возможность — выведение типа данных	60
Глава 3. Операторы	63
Арифметика	63
Простейшие операторы	63
Порядок выполнения операторов	64
Оператор присваивания	65
Оператор инкремента	65
Логично ли логическое сравнение	66
Сравнение чисел с плавающей точкой	67
Составные логические операторы	68
Тип выражения	69
Вычисление типа операции	69
Типы при присваивании	71
Глава 4. Управление потоком выполнения	73
Управление потоком выполнения	74
Оператор <code>if</code>	74
Инструкция <code>else</code>	77
Как избежать <code>else</code>	78
Вложенные операторы <code>if</code>	78
Конструкция <code>switch</code>	81
Циклы	83
Цикл <code>while</code>	83
Цикл <code>do...while</code>	88
Операторы <code>break</code> и <code>continue</code>	88
Цикл без счетчика	89
Правила области видимости	93
Цикл <code>for</code>	93
Пример	94
Зачем нужны разные циклы	94
Вложенные циклы	95
Оператор <code>goto</code>	96
Глава 5. Коллекционирование	99
Массивы <code>C#</code>	99
Зачем нужны массивы	99
Массив фиксированного размера	100
Массив переменного размера	102
Конструкция <code>foreach</code>	105
Сортировка массива объектов	106
Использование <code>var</code> для массивов	109
Коллекции <code>C#</code>	110

Синтаксис коллекций	111
Классы коллекций C#	111
Понятие <T>	111
Обобщенные коллекции	112
Использование списков	112
Использование словарей	115
Новинка: инициализаторы массивов и коллекций	117
Инициализация массивов	117
Инициализация коллекций	117
Новинка: использование множеств	118
Не используйте старые коллекции	122
Глава 6. Работа со строками в C#	123
Неизменяемость строк	124
Основные операции над строками	125
Сравнение строк	126
Метод Compare ()	126
Сравнение без учета регистра	129
Изменение регистра	129
Отличие строк в разных регистрах	130
Преобразование символов строки в символы верхнего или нижнего регистра	130
Цикл по строке	131
Поиск в строках	131
Как искать	132
Пуста ли строка	132
Получение введенной пользователем информации	132
Удаление пробельных символов	133
Анализ числового ввода	133
Обработка последовательности чисел	136
Объединение массива строк в одну строку	137
Управление выводом программы	138
Использование методов Trim() и Pad()	138
Использование функции конкатенации	140
Использование функции Split()	142
Форматирование строки	142
StringBuilder: эффективная работа со строками	146
ЧАСТЬ III. ИСПОЛЬЗОВАНИЕ ОБЪЕКТОВ	149
Глава 7. Немного о классах	151
Классы и объекты	151
Определение класса	152
Что такое объект	153
Доступ к членам объекта	153
Пример объектно-основанной программы	154
Отличие между объектами	156
Ссылки	156
Классы, содержащие классы	158

Статические члены класса	159
Определение константных членов-данных	160
Глава 8. Методы методов	161
Определение и использование метода	161
Использование методов в ваших программах	162
Аргументы метода	169
Передача аргументов методу	169
Передача методу нескольких аргументов	170
Соответствие определений аргументов их использованию	171
Перегрузка метода	172
Реализация аргументов по умолчанию	173
Передача в метод типов-значений	175
Возврат значений из метода	180
Возврат значения оператором <code>return</code>	181
Возврат значения посредством передачи по ссылке	182
Когда какой способ передачи использовать	182
Определение метода без возвращаемого значения	183
Глава 9. Пару слов об этом	187
Передача объекта в метод	187
Определение методов	189
Определение статического метода	189
Определение метода экземпляра	190
Полное имя метода	192
Обращение к текущему объекту	193
Ключевое слово <code>this</code>	194
Когда <code>this</code> используется явно	195
Что делать при отсутствии <code>this</code>	197
Новинка: методы расширения	199
Глава 10. Что такое объектно-ориентированное программирование	203
Объектно-ориентированная концепция № 1: абстракция	203
Приготовление блюд с помощью функций	204
Приготовление “объектно-ориентированных” блюд	204
Объектно-ориентированная концепция № 2: классификация	204
Зачем нужна классификация	205
Объектно-ориентированная концепция № 3: удобный интерфейс	206
Объектно-ориентированная концепция № 4: управление доступом	206
Поддержка объектно-ориентированных концепций в C#	207
ЧАСТЬ IV. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ	209
Глава 11. Класс — каждый сам за себя	211
Ограничение доступа к членам класса	211
Пример программы с использованием открытых членов	212
Прочие уровни безопасности	215

Зачем нужно управление доступом	215
Методы доступа	216
Пример управления доступом	217
Определение свойств класса	222
Новинка: дайте компилятору написать свойства за вас	224
Методы и уровни доступа	225
Конструирование объектов посредством конструкторов	225
Конструкторы, предоставляемые C#	225
Конструктор по умолчанию	227
Создание объектов	228
Выполнение конструктора в отладчике	230
Непосредственная инициализация объекта	233
Конструирование с инициализаторами	233
Новинка: инициализация объекта без конструктора	234
Перегрузка конструкторов	235
Устранение дублирования конструкторов	238
Фокусы с объектами	242
Глава 12. Наследование	243
Наследование класса	243
Зачем нужно наследование	245
Более сложный пример наследования	246
ЯВЛЯЕТСЯ или СОДЕРЖИТ	249
Отношение ЯВЛЯЕТСЯ	249
Доступ к BankAccount через содержание	250
Отношение СОДЕРЖИТ	251
Когда использовать отношение ЯВЛЯЕТСЯ, и когда — СОДЕРЖИТ	251
Поддержка наследования в C#	252
Изменение класса	252
Неверное преобразование времени выполнения	253
Ключевые слова <code>is</code> и <code>as</code>	253
Класс <code>object</code>	255
Наследование и конструктор	256
Вызов конструктора по умолчанию базового класса	256
Передача аргументов конструктору базового класса	257
Указание конкретного конструктора базового класса	259
Обновленный класс <code>BankAccount</code>	260
Глава 13. Полиморфизм	265
Перегрузка унаследованного метода	265
Простейший случай перегрузки метода	266
Различные классы, различные методы	266
Скрытие метода базового класса	267
Вызов методов базового класса	271
Полиморфизм	273
Что неверно в стратегии использования объявленного типа	274
Использование <code>is</code> для полиморфного доступа к скрытому методу	275
Объявление метода виртуальным	276
Трюк “один за всех, все за одного”	279

Визитная карточка класса: метод ToString ()	279
Абстракционизм в C#	279
Разложение классов	280
Голая концепция, выражаемая абстрактным классом	284
Как использовать абстрактные классы	284
Создание абстрактных объектов невозможно	286
Опечатывание класса	287
Глава 14. Интерфейсы	289
Что значит МОЖЕТ_ИСПОЛЬЗОВАТЬСЯ_КАК	289
Что такое интерфейс	290
Реализация интерфейса	291
Именование интерфейсов	292
Зачем в C# включены интерфейсы	292
Наследование и реализация интерфейса	292
Преимущества интерфейсов	293
Использование интерфейсов	294
Тип, возвращаемый методом	294
Базовый тип массива или коллекции	294
Более общий тип ссылки	295
Использование предопределенных типов интерфейсов C#	295
Пример программы, использующей отношение	
МОЖЕТ_ИСПОЛЬЗОВАТЬСЯ_КАК	296
Создание собственного интерфейса	296
Реализация интерфейса IComparable<T>	297
Сборка воедино	298
Унификация иерархий классов	302
Что скрыто за интерфейсом	305
Наследование интерфейсов	307
Использование интерфейсов для внесения изменений	
в объектно-ориентированные программы	308
Гибкие зависимости через интерфейсы	309
Абстрактный или конкретный? Когда следует использовать	
абстрактный класс, и когда — интерфейс	309
Осуществление отношения СОДЕРЖИТ с помощью интерфейсов	310
Шаблон проектирования Strategy	311
Чем полезны шаблоны	312
ЧАСТЬ V. C# 3.0	315
Глава 15. Делегирование событий	317
Звонок домой — проблема обратного вызова	317
Что такое делегат	318
Примеры передачи кода	320
Очень простой первый пример	320
Пример из реального мира	322
Обзор большего примера	322
Создание приложения	322

Знакомимся с кодом	324
Жизненный цикл делегата	326
Анонимные методы	328
События C#	329
Шаблон проектирования <code>Observer</code>	329
Что такое событие. Публикация и подписка	330
Как издатель оповещает о своих событиях	330
Как подписаться на событие	331
Как опубликовать событие	331
Как передать обработчику события дополнительную информацию	332
Рекомендованный способ генерации событий	333
Как наблюдатели “обрабатывают” событие	334
Глава 16. Лямбда-выражения	335
Коллекции и задача “для каждого”	335
Новинка: применение лямбда-выражений	335
Способ “для каждого” в C#	336
Заготовка: данные для примеров	337
Несколько примеров с методом <code>Find()</code>	337
Выполнение действий: примеры с <code>ForEach()</code>	339
Лямбда-выражение на основе пользовательского делегата	342
Резюмируя примеры	343
Синтаксис лямбда-выражений	343
Применение лямбда-выражений для произвольного делегата	345
Глава 17. Выражения запросов	347
Обзор возможностей C# 3.0	347
Что же такое запрос	348
Почему важны запросы, интегрированные в язык	349
Общий вид запроса C#	350
Более пристальный взгляд на LINQ	350
Запросы в C# 3.0	351
Три самых важных оператора запроса	352
Что означает отложенное выполнение запроса	354
Немедленное выполнение запроса	355
Что могут запросы	355
Фильтрация	356
Пример 1: кто из заказчиков живет в Колорадо?	356
Выбор и группирование	358
Пример 2: последовательность чисел, увеличенных на 2	358
Пример 3: имя и телефонный номер каждого заказчика	359
Использование анонимных типов	359
Пример 4: преобразование чисел в слова	361
Пример 5: группирование имен по их первой букве	362
Пример 6: вывод некоторых групп	364
Пример 7: группирование товаров по категориям	365
Сортировка	366
Пример 8: упорядочение групп из примеров 5 и 6	367
Пример 9: сортировка в группах	367

Пример 10: еще одна двухуровневая сортировка	368
Пример 11: сортировка в убывающем порядке	368
Метод запроса с методами	369
Пример 12: написание запросов с методами вместо ключевых слов	370
О поездах для новичков	370
Передышка для “стариков” закончена	372
Обобщенные методы	372
Что внутри Where ()	372
Пример 13: имя и телефонный номер каждого заказчика	373
Подсчет, суммирование, усреднение и прочее	374
Пример 14: подсчет типов товара на складе	374
Пример 15: суммирование элементов массива	374
Пример 16: суммарная стоимость заказа	375
Пример 17: вычисление факториала 9	375
Работа с объектами как с коллекциями	376
ЧАСТЬ VI. ВЕЛИКОЛЕПНЫЕ ДЕСЯТКИ	377
Глава 18. Десять наиболее распространенных ошибок компиляции	379
The name ‘memberName’ does not exist in the class or namespace ‘className’	379
Cannot implicitly convert type ‘x’ into ‘y’	381
‘className.memberName’ is inaccessible due to its protection level	383
Use of unassigned local variable ‘n’	384
Unable to copy the file ‘programName.exe’ to ‘programName.exe’. The process cannot...	384
‘subclassName.methodName’ hides inherited member ‘baseclassName.methodName’.	
Use the new keyword if hiding was intended	385
‘subclassName’ : cannot inherit from sealed class ‘baseclassName’	386
‘className’ does not implement interface member ‘methodName’	386
‘methodName’ : not all code paths return a value	387
} expected	388
ЧАСТЬ VII. ДОПОЛНИТЕЛЬНЫЕ ГЛАВЫ	389
Глава 19. Пространства имен и библиотеки	391
Разделение одной программы на несколько исходных файлов	391
Разделение единой программы на сборки	393
Выполнимый файл или библиотека	393
Сборки	393
Выполнимые файлы	394
Библиотеки классов	394
Объединение классов в библиотеки	395
Создание проекта библиотеки классов	395
Создание автономной библиотеки классов	396
Добавление второго проекта к существующему решению	396
Создание классов для библиотеки	398
Использование драйвера для тестирования библиотеки	398
Использование библиотеки классов в программе	399
Дополнительные ключевые слова для управления доступом	400

internal: строим глазки ЦРУ	400
protected: поделимся с подклассами	402
protected internal: более изощренная защита	404
Размещение классов в пространствах имен	405
Объявление пространств имен	406
Пространства имен и доступ	408
Использование полностью квалифицированных имен	409
Директива using	410
Пример использования полностью квалифицированных имен	411
Глава 20. Эти исключительные исключения	415
Старый способ обработки ошибок	415
Возврат индикатора ошибки	417
Чем плохи коды ошибок	421
Использование механизма исключений для сообщения об ошибках	422
О try-блоках	424
О catch-блоках	424
О finally-блоках	424
Что происходит при генерации исключения	425
Генерация исключений	427
Для чего нужны исключения	427
Пример	427
Что делает этот пример “исключительным”	429
Трассировка стека	430
Использование нескольких catch-блоков	431
Планирование стратегии обработки ошибок	433
Вопросы, помогающие при планировании	433
Советы по написанию кода с хорошей обработкой ошибок	434
Анализ возможных исключений метода	436
Как выяснить, какие исключения генерируются теми или иными методами	438
Последний шанс перехвата исключения	438
Глава 21. Рыбалка в потоке	441
Где водится рыба: файловые потоки	441
Потоки	441
Читатели и писатели	442
Использование StreamWriter	443
Пример использования потока	444
Как это работает	446
Наконец-то мы пишем!	449
Использование конструкции using	450
Использование StreamReader	453
Еще о читателях и писателях	457
Прочие виды потоков	458
Глава 22. Значения и ссылки	461
Структуры C# и их отличия от классов	461
Структуры C#	462
Конструктор структуры	463

Методы структур	464
Пример применения структуры	465
Когда использовать структуры, а когда — классы	468
Предопределенные типы структур	468
Перечисления — группировка именованных констант	469
Унификация системы типов	469
Упаковка типов-значений	472
Как избежать упаковки и распаковки	472
Глава 23. Использование интерфейса Visual Studio	475
Окна, окна, окна...	475
Настройка расположения окон	476
Состояния окон	476
Соккрытие окон	478
Перестановка окон	479
Наложение окон	481
Дополнительные возможности Visual Studio	482
Просмотр диаграммы классов	482
Рефакторинг кода	482
Стандартные заготовки кода	484
Автоматизация Visual Studio при помощи макросов	484
Работа с Solution Explorer	485
Как упростить жизнь с помощью проектов и решений	486
Отображение проекта	487
Навигация по исходному тексту	490
Добавление класса	491
Завершение демонстрационной программы	492
Преобразование классов в программу	495
Помогите мне!	496
Отладка	496
Жучки в программе: а дустом не пробовали?	497
Пошаговая отладка	498
Главное — вовремя остановиться	502
Стек вызовов	504
Я сделал это!	508
Глава 24. Обобщенность	509
Обобщенность в C#	509
Обобщенные классы безопасны	509
Обобщенные классы эффективны	510
Создание собственного обобщенного класса	511
Очередь с приоритетами	511
Распаковка пакета	516
Метод <code>Main()</code>	517
Написание обобщенного кода	518
Обобщенная очередь с приоритетами	519
Использование простого необобщенного класса фабрики	521
Незавершенные дела	523

Обобщенные методы	525
Обобщенные методы в необобщенных классах	527
Обобщенные методы в обобщенных классах	528
Ограничения для обобщенного метода	528
Обобщенные интерфейсы	529
Объявление обобщенного интерфейса	529
Написание класса, реализующего обобщенный интерфейс	529
Интерфейсы на примере	530
Обобщенное делегирование	533
Глава 25. Работа с коллекциями	537
Обход каталога файлов	537
Написание собственного класса коллекции: связанный список	543
Пример связанного списка	544
Зачем нужен связанный список	553
Обход коллекций: итераторы	553
Доступ к коллекции: общая задача	554
Использование <code>foreach</code>	555
Обращение к коллекциям как к массивам: индексаторы	557
Формат индексатора	557
Пример программы с использованием индексатора	557
Блок итератора	561
Итерация месяцев	566
Что такое коллекция	567
Синтаксис итератора	568
Блоки итераторов произвольного вида и размера	569
Где надо размещать итераторы	572
Предметный указатель	581