КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
НАБЕРЕЖНОЧЕЛНИНСКИЙ ИНСТИТУТ

С.В. Булатова

ПРОГРАММИРОВАНИЕ. ЧАСТЬ 1

Учебно-методическое пособие

НАБЕРЕЖНЫЕ ЧЕЛНЫ
2023

УДК 004.432C#(076.5)
ББК 32.973.22я73-5
Б90

Печатается по рекомендации Учебно-методической комиссии
Набережночелнинского института (филиала)
Казанского (Приволжского) федерального университета
(протокол № 3 от 23 марта 2023 г.)

Рецензенты:
канд. техн. наук, доцент кафедры автоматизации и управления **И.П. Балабанов**;
канд. техн. наук, доцент кафедры информационных систем **Е.В. Зубков**

Булатова С.В.

Б90 **Программирование. Часть 1.:** учебно-методическое пособие / С.В. Булатова. – Набережные Челны : Изд.-полигр. центр Набережночелнинского института КФУ, 2023. – 19 с.

Учебно-методическое пособие предназначено для изучения языка программирования C#. Язык C# как средство обучения программированию обладает рядом несомненных достоинств. Он хорошо организован, строг, большинство его конструкций логичны и удобны. По рассмотренным темам, приведённым в учебно-методическом пособии, представлен теоретический материал, рассмотрены примеры задач и даны задания для самостоятельного решения.

Для студентов направления подготовки «Программная инженерия».

УДК 004.432C#(076.5)
ББК 32.973.22я73-5

ВВЕДЕНИЕ

Язык C# как средство обучения программированию обладает рядом несомненных достоинств. Развитые средства диагностики и редактирования кода делают процесс программирования приятным и эффективным. Мощная библиотека классов платформы .NET берет на себя массу рутинных операций, что даёт возможность решать более сложные задачи, используя готовые «строительные блоки». Все это позволяет расценивать C# как перспективную замену языков при обучении программированию.

Немаловажно, что C# является не учебным, а профессиональным языком, предназначенным для решения широкого спектра задач, и в первую очередь — в быстро развивающейся области создания распределенных приложений. Поэтому базовый курс программирования, построенный на основе языка C#, позволит студентам быстрее стать востребованными специалистами - профессионалами.

Мощь языка C# имеет и оборотную сторону: во-первых, он достаточно требователен к ресурсам компьютера, во-вторых, для осмысленного написания простейшей программы, вычисляющей, «сколько будет дважды два», требуется изучить достаточно много материала, но многочисленные достоинства языка и платформы .NET перевешивают все недостатки. По рассмотренным темам, приведённым в лабораторном практикуме, представлен теоретический материал, рассмотрены примеры задач и даны задания для самостоятельного решения.

ЛАБОРАТОРНАЯ РАБОТА № 1
ПЕРЕМЕННЫЕ, ТИПЫ ДАННЫХ, КОНСТАНТЫ. АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОПЕРАЦИИ.

1. Цель работы. Получение навыков в разработке программ на языке C#.

2. Общие сведения

Переменная – это именованная область памяти. В переменную можно записывать данные и считывать. Данные, записанные в переменной, называются значением переменной. C# – язык жесткой типизации. Каждая переменная должна быть определенного типа данных. Ниже в таблице наведены встроенные типы данных языка C#:

Тип	Область значений	Размер
sbyte	-128 до 127	Знаковое 8-бит целое
byte	0 до 255	Беззнаковое 8-бит целое
char	U+0000 до U+ffff	16-битовый символ Unicode
bool	true или false	1 байт*
short	-32768 до 32767	Знаковое 16-бит целое
ushort	0 до 65535	Беззнаковое 16-бит целое
int	-2147483648 до 2147483647	Знаковое 32-бит целое
uint	0 до 4294967295	Беззнаковое 32-бит целое
long	-9223372036854775808 до 9223372036854775807	Знаковое 64-бит целое
ulong	0 до 18446744073709551615	Беззнаковое 64-бит целое
float	±1,5*10-45 до ±3,4*1033	4 байта, точность — 7 разрядов
double	±5*10-324 до ±1,7*10306	8 байтов, точность — 16 разрядов
decimal	(-7,9 * 1028 до 7,9 * 1028) / (100–28)	16 байт, точность — 28 разрядов

Для того, чтобы использовать переменную, ее сначала нужно объявить:

```
static void Main(string[] args)
{
  int a; // объявляем переменную a типа int
  a = 5; // записываем в переменную a число 5
  int b, c; // объявить можно сразу несколько переменных через запятую
  bool d; // объявляем переменную d типа bool
  d = true; // записываем в переменную d значение true (истина)
  long e = 10; // при объявлении переменной можно сразу же задавать ей значение, это называется инициализацией
  float f = 5.5f; // чтобы записать число с плавающей точкой типа float, нужно после значения добавлять суффикс f.
  char g = 'g'; // объявление символьной переменой g с ее инициализацией значением символа 'g'
}
```

При использовании переменной, в которую не было записано значение, компилятор выдаст ошибку "Use of unassigned local variable [variableName]".

Для того, чтобы объявить константу, перед обычным объявлением переменной нужно добавить ключевое слово const:

```
static void Main(string[] args)
{
  const int months = 12; // объявлениеконстанты
  months = 13; // ошибкакомпиляции
}
```

var сохраняет принцип строгой типизации в C#. Это означает, что после того, как для переменной уже был определен тип, в нее нельзя записать данные другого типа:

```
static void Main(string[] args)
{
  var number = 5;
  number = "some text"; // ошибка, number определенкак int
}
```

Все операции делятся на два типа: унарные и бинарные. К унарным относятся операции, в которых участвует один операнд. В бинарных операциях – два операнда. Список бинарных арифметических операций приведен в таблице:

Операция	Запись
Сложение	a + b
Вычитание	a - b
Деление	a / b
Умножение	a * b
Нахождение остатка от деления	a % b

Унарных арифметических операторов в C# есть всего два: инкрементация «++» и декрементация «--»; Инкрементация увеличивает операнд на единицу, а декрементация – уменьшает на единицу.

```
static void Main(string[] args)
{
  int a = 0, b = 5;
  a++; // a=1;
  b--; // b=4
}
```

Класс Math

Для возведения числа в степень, используется функция Pow([число], [степень]);

```
static void Main(string[] args)
{
  float a, b = 9;
  a = (float) Math.Pow(b, 2);
  Console.WriteLine(a);
  Console.ReadKey();
}
```

Для нахождения квадратного корня служит функция Sqrt([число]); возвращаемый результат также в типе данных double:

```
static void Main(string[] args)
{
```

```
  double a, b = 9;
  a = Math.Sqrt(b);
  Console.WriteLine(a); // выводитнаэкранчисло 3
  Console.ReadKey();
}
```

В C# есть следующие логические операторы:

=! – оператор «НЕ» является унарным и возвращает противоположное значение операнда.

```
static void Main(string[] args)
{
  bool a, b = true, c = false;
  a = !b; // a = false
  a = !c; // a = true
}
```

|| – оператор «ИЛИ» является бинарным и возвращает false только тогда, когда оба операнда равны false, в остальных случаях результат будет true;

```
static void Main(string[] args)
{
  bool a, bTrue = true, bFalse = false;
  a = bFalse || bFalse; // a = false
  a = bFalse || bTrue; // a = true
 }
```

&& – оператор «И» является бинарным и возвращает true только тогда, когда оба операнда равны true, в остальных случаях результат будет false;

```
static void Main(string[] args)
{
  bool a, bTrue = true, bFalse = false;
  a = bFalse && bFalse; // a = false
  a = bFalse && bTrue; // a = false
 }
```

К операторам сравнения относятся:

Оператор	Название
>	больше
<	меньше
>=	больше или равно
<=	меньше или равно
==	равно
!=	неравно

```
static void Main(string[] args)
{
  bool a;
```

```
  int b = 2, c = 3, d = 2;
  a = b > c; // a = false
  a = b == d; // a = true
  a = b != c; // a = true
}
```

3. Постановка задачи

Разработать блок-схему и программный код, согласно своему варианту, объявите несколько переменных различных типов, примените явное и неявное преобразование. Создайте константную переменную, попробуйте изменить ее значение.

1. Дана длина ребра куба. Найти объем куба и площадь его боковой поверхности.
2. Даны катеты прямоугольного треугольника. Найти его гипотенузу и площадь.
3. Определить периметр и площадь правильного n-угольника, описанного около окружности радиуса r.
4. Дана сторона равностороннего треугольника. Найти площадь этого треугольника.
5. Известна длина окружности. Найти площадь круга, ограниченного этой окружностью.
6. Найти площадь кольца, внутренний радиус которого R1, а внешний R2.
7. Треугольник задан величинами своих углов и радиусом описанной окружности. Найти стороны треугольника.
8. Найти площадь равнобочной трапеции с основаниями a и b и углом альфа при большем основании a.
9. Треугольник задан длинами сторон. Найти длины высот.
10. Вычислить расстояние между двумя точками с координатами x1, y1 и x2, y2.
11. Треугольник задан длинами сторон. Найти радиусы вписанной и описанной окружностей.
12. Треугольник задан координатами своих вершин. Найти периметр треугольника.
13. Треугольник задан длинами сторон. Найти длины меридиан.
14. Треугольник задан координатами своих вершин. Найти площадь треугольника.
15. Треугольник задан длинами сторон. Найти площадь треугольника по формуле Герона.

4. Порядок выполнения работы

В главном меню системы программирования выберите команду File→New→Project или щелкните на соответствующей инструментальной кнопке. В окне New Project выберите пункт Console Application и щелкните на кнопке OK.

5. Содержание отчета

1. Тема лабораторной работы
2. Цель работы
3. Блок-схема
4. Программный код и результат работы программы
5. Вывод о проделанной лабораторной работе

ЛАБОРАТОРНАЯ РАБОТА № 2
УСЛОВНЫЕ ОПЕРАТОРЫ В C#. ТЕРНАРНЫЙ ОПЕРАТОР.

1. Цель работы. Получение навыков в разработке программ с применением условного и тернарного операторов на языке C#.

2. Общие сведения

Условные операторы служат для ветвления программы. В зависимости от некоторого условия выполняется тот или другой набор команд.

Оператор «if-else»

Данный оператор имеет следующую структуру:

```
if ([условное выражение])
{
  Блок кода, который нужно выполнить при удовлетворении условия,   [условное выражение] = true (истина)
}
else
{
  Блок кода, который нужно выполнить при неудовлетворении условия,   [условное выражение] = false (ложь)
}
```

Часть else не является обязательной и может отсутствовать. Пример использования оператора «if-else» в программе, которая проверяет вводимое число на чётность:

```
static void Main(string[] args)
{
  int a;
  Console.WriteLine("Введите число:");
  a = Convert.ToInt32(Console.ReadLine()); // вводим данные с клавиатуры*
  if (a % 2 == 0) //проверяем число на чётность путем нахождения остатка от деления числа на 2
  {
    Console.WriteLine("Число " + a + " - чётное");
  }
  else
  {
    Console.WriteLine("Число " + a + " - нечётное");
  }
  Console.ReadKey();
}
```

Функция Console.ReadLine() позволяет ввести данные с клавиатуры. Данные вводятся как строка, а так как нужно число, мы преобразовываем ее в числовой тип. Для преобразования мы используем функцию Convert.ToInt32().

Если после if или else необходимо выполнить лишь одну команду, фигурные скобки можно опускать:

```
if ([условное выражение])
[команда1] // команда1 выполнится лишь если условное выражение истинно
[команда2]// команда2 выполнится в любом случае
```

Оператор if может иметь несколько условий:

```
if ([логическое выражение1])
{блок1}
else if ([логическое выражение2])
{блок2}
else
{блок3}
```

Пример программы, которая определяет, какое из двух введенных чисел больше:

```
static void Main(string[] args)
{
  int a, b;
  Console.WriteLine("Введите первое число:");
  a = Convert.ToInt32(Console.ReadLine());
  Console.WriteLine("Введите второе число:");
  b = Convert.ToInt32(Console.ReadLine());
  if (a > b)
    Console.WriteLine("Первое число больше второго");
  else if (a < b)
                Console.WriteLine("Второе число больше первого");
        else
                Console.WriteLine("Числа равны");
  Console.ReadKey();
}
```

Логическое выражение может быть сложнее. Здесь и используются логические операторы «!», «||» и «&&».

В некоторых случаях удобно использовать условный оператор «switch» вместо «if-else». Он имеет следующую структуру:

```
switch (выражение)
{
  case значение1:
    блок1;
    break;
  case значение2:
    блок2;
    break;
    ...
  case значениеN:
    блокN;
    break;
  default:
    блокN+1;
```

```
    break;
}
```

Пример программы с использованием switch:

```
static void Main(string[] args)
{
  int a;
  Console.WriteLine("Введите порядковый номер дня недели:");
  a = Convert.ToInt32(Console.ReadLine());
  switch (a)
  {
    case 1:
      Console.WriteLine("Понедельник");
      break;
    case 2:
      Console.WriteLine("Вторник");
      break;
    case 3:
      Console.WriteLine("Среда");
      break;
    case 4:
      Console.WriteLine("Четверг");
      break;
    case 5:
      Console.WriteLine("Пятница");
      break;
    case 6:
      Console.WriteLine("Суббота");
      break;
    case 7:
      Console.WriteLine("Воскресенье");
      break;
    default :
      Console.WriteLine("Ошибка");
      break;
    }
    Console.ReadKey();
}
```

Тернарный оператор «?:»

Этот оператор используется для сокращения объема кода. Им можно заменять простые по сложности операторы if-else. Тернарный оператор имеет такую структуру:

логическое выражение? выражение1 : выражение2

Пример использования тернарного оператора «?:» в той же программе для проверки числа на чётность:

```
static void Main(string[] args)
{
  int a;
  Console.WriteLine("Введитечисло:");
  a = Convert.ToInt32(Console.ReadLine());
  Console.WriteLine( a % 2 == 0 ? "Число чётное" : "Число нечётное" );
  Console.ReadKey();
}
```

«?:» также можно использовать для присваивания значений. Пример программы, которая находит большее число из двух вводимых:

```
static void Main(string[] args)
{
  int a, b, max;
  Console.WriteLine("Введитепервоечисло:");
  a = Convert.ToInt32(Console.ReadLine());
  Console.WriteLine("Введитевтороечисло:");
  b = Convert.ToInt32(Console.ReadLine());
  max = a > b ? a : b;
}
```

3. Постановка задачи

Разработать блок-схему и программный код на языке C#, согласно своему варианту.

Задание №1.

1. Дано действительное число a. Составить программу, которая определяет отрицательность или положительность данного числа и выводит соответствующее сообщение.

2. Даны два числа. Проверить какое из чисел минимальное и насколько оно меньше второго числа.

3. Дано выигрышное число. Определить больше оно 1000 или меньше.

4. Даны два целых числа. Определить, какое из чисел максимальное.

5. Даны два целых числа. Определить, какое из чисел минимальное.

6. Даны два числа проверить равенство чисел и выдать соответствующее сообщение.

7. Даны два числа. Проверить какое из чисел максимальное и насколько оно больше второго числа.

8. Проснувшись утром, школьник почувствовал недомогание. Проверив температуру, он решил: если температура меньше 37 градусов пойду в школу, иначе вызову врача.

9. Дано целое число. Проверить может ли являться число радиусом окружности.

10. Даны два числа. Найдите среди них максимальный и возведите это число в квадрат.

11. Даны два числа. Определить которое из них минимальное и найти корень из этого числа.

12. Дано число. Определить существует ли корень из этого числа.

13. Даны координаты точки. Определить какой координатной четверти она принадлежит.

14. Даны два числа. Определить среди них минимальный. И если он больше нуля, то найти корень из этого числа, иначе возвести его в квадрат.

15. Даны два числа. Определить которое из них максимальное и возвести число в n степень.

Задание №2.

1. Даны три действительных числа. Найти среди данных чисел максимальный и возвести в 6-ую степень.

2. Даны три действительных числа. Найти среди данных чисел минимальный и максимальный. Определить разность этих чисел.

3. Даны три действительных числа. Найти среди данных чисел минимальный и определить синус этого числа.

4. Даны три действительных числа. Проверить есть ли среди них одинаковые числа.

5. Даны три действительных числа. Определить среднее число и вывести соответствующее сообщение об этом.

6. Даны три целых числа. Определить есть ли среди данных чисел нули.

7. Даны три целых числа. Проверить являются ли эти числа пифагоровыми.

8. Даны три целых числа. Определить максимальное и минимальное среди этих чисел и найти сумму минимального и максимального.

9. Даны три числа. Определить могут ли эти числа быть сторонами треугольника.

10. Даны три числа. Определить количество чисел равных 5.

11. Даны три целых числа. Определить среди них минимальный и максимальный и возвести минимальное число в максимальную степень.

12. Даны три целых. Посчитайте среди данных чисел количество положительных и отрицательных.

13. Даны три числа. Если число отрицательное, то найти модуль этого числа и найти сумму этих чисел.

14. Даны три числа. Если среди данных чисел нет нулей, то найти произведение этих чисел, иначе вывести соответствующее сообщение.

15. Даны три действительных числа. Проверить есть ли среди них четные числа.

4. Порядок выполнения работы

В главном меню системы программирования выберите команду File→New→Project или щелкните на соответствующей инструментальной кнопке. В окне New Project выберите пункт Console Application и щелкните на кнопке OK.

5. Содержание отчета

1. Тема лабораторной работы
2. Цель работы
3. Блок-схема
4. Программный код и результат работы программы
5. Вывод о проделанной лабораторной работе

ЛАБОРАТОРНАЯ РАБОТА № 3
ЦИКЛЫ В C#. ОПЕРАТОРЫ BREAK И CONTINUE

1. Цель работы. Получение навыков в разработке программ с применением циклов на языке C#.

2. Общие сведения

Циклы служат для многократного повторения некоторого фрагмента кода.

Цикл for

Этот цикл используется тогда, когда наперёд известно, сколько повторений нужно сделать. Он имеет следующую структуру:

```
for (инициализация счётчика; условие продолжения; итерация)
{
  //блок кода, который будет повторяться
}
```

Пример программы, которая находит и выводит на экран сумму элементов массива:

```
static void Main(string[] args)
{
  int[] numbers = { 4, 7, 1, 23, 43 };
  int s = 0;
  for (int i = 0; i < numbers.Length; i++)
  {
    s += numbers[i];
  }
  Console.WriteLine(s);
  Console.ReadKey();
}
```

Счетчик можно изменять не только на единицу. Пример программы, которая выводит чётные числа (по число 50):

```
for (int i = 0; i <= 50; i+=2) //выполнится 26 раз
{
  Console.WriteLine(i);
}
```

Цикл while

Слово while переводится, как «пока», что хорошо его характеризует. Он продолжает выполнятся до тех пор, пока «истинно» некоторое условие. Он имеет такую структуру:

```
while (условие продолжения)
{
  //блок кода, который будет повторяться
}
```

Сначала проверяется условие, а дальше выполняется блок кода. Пример той же программы, которая выводит на экран числа 0, 1, 2, 3, 4:

```
int i = 0;
while (i < 5)
{
```

```
  Console.WriteLine(i);
  i++;
}
```

Цикл может выполнятся «вечно», если задать всегда истинное условие:

```
while (true)
{
  Console.WriteLine("Вечныйцикл");
}
```

Цикл do-while

Этот тот же цикл while, только здесь сначала выполняется блок кода, а уже потом идет проверка условия. Это гарантирует хотя бы один проход цикла.

```
do
{
//блок кода, который будет повторяться
}
while (условие продолжения);
```

Пример программы, которая не завершит работу, пока с клавиатуры не введут число 5:

```
static void Main(string[] args)
{
  int number;
  do
  {
    Console.WriteLine("Введитечисло 5");
    number = Convert.ToInt32(Console.ReadLine());
  }
  while (number != 5);
}
```

Оператор break

Из любого цикла можно досрочно выйти, используя оператор break. Пример программы, которая проверяет, есть ли в массиве число кратное 13-ти. Найдя такое число, нет смысла дальше проверять остальные элементы массива, и здесь мы используем оператор break:

```
static void Main(string[] args)
{
  int[] numbers = { 4, 7, 13, 20, 33, 23, 54 };
  bool b = false;
  for (int i = 0; i < numbers.Length; i++)
  {
    if (numbers[i] % 13 == 0)
    {
      b = true;
      break;
    }
  }
```

```
  Console.WriteLine(b ? "В массиве есть число кратное 13" : "В массиве нет числа кратного 13");
  Console.ReadKey();
}
```

Оператор continue

Данный оператор позволяет перейти к следующей итерации, не завершив до конца текущую. Пример программы, которая находит сумму нечетных элементов массива:

```
static void Main(string[] args)
{
  int[] numbers = { 4, 7, 13, 20, 33, 23, 54 };
  int s = 0;
  for (int i = 0; i < numbers.Length; i++)
  {
    if (numbers[i] % 2 == 0)
      continue; //переход к следующей итерации
    s += numbers[i];
    }
  Console.WriteLine(s);
  Console.ReadKey();
}
```

3. Постановка задачи

Разработать блок-схему и программный код на языке C#, согласно своему варианту.

Задание №1

1. Дана последовательность из n целых чисел. Составить программу нахождения суммы положительных чисел кратных 3.

2. Дана последовательность из n целых чисел. Составить программу нахождения произведения отрицательных чисел кратных 3 и 7.

3. Дана последовательность из n целых чисел. Составить программу нахождения разности положительных чисел кратных 5 и 7.

4. Дана последовательность из n целых чисел. Составить программу нахождения суммы нечетных неотрицательных элементов последовательности.

5. Дана последовательность из n целых чисел. Составить программу нахождения синуса чисел кратных 2.

6. Дана последовательность из n целых чисел. Составить программу нахождения тангенса чисел кратных 3 и 5.

7. Дана последовательность из n целых чисел. Составить программу нахождения суммы чисел, являющихся полными квадратами.

8. Дана последовательность из n целых чисел. Составить программу нахождения количества чисел кратных 3.

9. Дана последовательность из n целых чисел. Составить программу нахождения количества четных чисел.

10. Дана последовательность из n целых чисел. Составить программу нахождения количества чисел кратных 5 и 7.

11. Дана последовательность из n чисел. Составить программу нахождения косинуса неотрицательных, нечетных чисел кратных 3.

12. Дана последовательность из n целых чисел. Составить программу нахождения суммы отрицательных чисел кратных 3.

13. Дана последовательность из n целых чисел. Составить программу нахождения произведения нечетных чисел кратных 3, 6, 9.

14. Дана последовательность из n целых чисел. Составить программу нахождения суммы чисел по формуле cos(x)+sin(x) кратных 5.

15. Дана последовательность из n целых чисел. Составить программу нахождения количества четных и нечетных элементов последовательности.

Задание №2.

1. Дано целое n-значное число. Составить программу нахождения суммы четных цифр числа.

2. Дано целое n-значное число. Составить программу нахождения произведения нечетных цифр числа.

3. Дано целое n-значное число. Составить программу подсчета количества четных цифр числа.

4. Дано целое n-значное число. Составить программу подсчета количества цифр числа.

5. Дано целое n-значное число. Составить программу подсчета количества цифр 5, в записи числа.

6. Дано целое n-значное число. Составить программу подсчета количества цифр 1, в записи числа.

7. Дано целое n-значное число. Составить программу подсчета количества нулей, в записи числа.

8. Дано целое n-значное число. Составить программу подсчета количества цифр 7, в записи числа.

9. Дано целое n-значное число. Составить программу подсчета количества цифр числа кратные 3.

10. Дано целое n-значное число. Составить программу подсчета количества цифр числа.

11. Дано целое n-значное число. Составить программу нахождения произведения нечетных цифр числа.

12. Дано целое n-значное число. Составить программу подсчета количества цифр 7, в записи числа.

13. Дано целое n-значное число. Составить программу подсчета количества цифр 2, в записи числа.

14. Дано целое n-значное число. Составить программу нахождения четных цифр в записи числа.

15. Дано целое n-значное число. Составить программу подсчета количества цифр 6, в записи числа.

4. Порядок выполнения работы

В главном меню системы программирования выберите команду File→New→Project или щелкните на соответствующей инструментальной кнопке. В окне New Project выберите пункт Console Application и щелкните на кнопке OK.

5. Содержание отчета

1. Тема лабораторной работы
2. Цель работы
3. Блок-схема
4. Программный код и результат работы программы
5. Вывод о проделанной лабораторной работе

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Тузовский, А.Ф. Объектно-ориентированное программирование: учеб. пособие для прикладного бакавлариата / А.Ф. Тузовский. – М.: Юрайт, 2018. – 206 с. – (Университеты России). – ISBN 978-5-534-00849-4.

2. Медведев, М.А. Программирование на C#: учеб. пособие / М.А. Медведев, А.Н. Медведев. – Екатеринбург : Изд-во Урал. ун-та, 2015. – 64 с. – ISBN 978-5-7996-1561-1.

Оглавление

Учебное издание

Булатова Светлана Владимировна

ПРОГРАММИРОВАНИЕ. ЧАСТЬ 1

Учебно-методическое пособие

Редактор
Г.Ф. Таипова

Компьютерная верстка
А.А. Фахуртдинова

Подписано в печать 22.05.2023.
Бумага офсетная. Печать ризографическая.
Формат 60×84 1/16. Гарнитура «Times New Roman».
Усл. печ. л. 1,16. Уч.-изд. л. 0,62. Тираж 50 экз. Заказ 1777-533

Отпечатано в издательско-полиграфическом центре
Набережночелнинского института
Казанского (Приволжского) федерального университета

423810, г. Набережные Челны, Новый город, проспект Мира, 68/19
тел./факс (8552) 39-65-99 e-mail: ic-nchi-kpfu@mail.ru