



Модуль School базовых алгоритмов информатики

Осипов А.В.

**Конференция «Использование системы программирования PascalABC.NET
в обучении программированию», 13.11.2021**

**Южный федеральный университет
г. Ростов-на-Дону**



Назначение: реализация алгоритмов, часто встречающихся в школьных задачах.

Форматы вызова: в виде подпрограммы и для записи в точечной нотации.

Исходный код: PascalABC.NET, открытый, в составе стандартной поставки.

Лицензия: открытая, совпадает с лицензией на весь программный продукт.

Подключение: uses School;

2

Конференция «Использование системы программирования PascalABC.NET в обучении программированию», 13.11.2021

Южный федеральный университет
г. Ростов-на-Дону



Библиотека разбита на 12 групп по функциональному назначению реализованных алгоритмов.

Библиотека содержит подпрограммы и методы расширения некоторых стандартных классов PascalABC.NET.

3

Конференция «Использование системы программирования PascalABC.NET в обучении программированию», 13.11.2021

**Южный федеральный университет
г. Ростов-на-Дону**



Подпрограмма – именованный или идентифицированный иным образом фрагмент программного кода, к которому можно многократно обращаться. Она содержит описание определённого набора действий. Подпрограмма – общее название для процедур и функций.



Функция – разновидность подпрограммы, возвращающая некоторое значение, которое затем используется в выражении на месте, где был указан вызов функции.

`Sin(2 * x - 1) + Cos(0.4 * x)`

`Min(a, b, c) + Max(a + b, 5 * c - 1)`

`Sqrt(Pi / 2 - 0.28)`

5

Конференция «Использование системы программирования PascalABC.NET в обучении программированию», 13.11.2021

**Южный федеральный университет
г. Ростов-на-Дону**



Процедура – разновидность подпрограммы, не возвращающая значения.

```
Print(a * x * x + b * x * c);  
Read(a, b, c);  
Delete(s, 5, 2);
```



Метод представляет собой подпрограмму, оформленную внутри класса или записи *record*.
Метод расширения (или просто «расширение») – разновидность метода, написанная в дополнение к уже имеющемуся классу и находящаяся вне его кода.

- a.Sort
- b.IsPrime
- c.ToBase(12)

1. Перевод десятичного числа n в систему счисления по основанию k ($k = 2..36$)



- функция $Bin(n)$ возвращает двоичное (binary) представление значения десятичного числа n в виде строки;
- функция $Oct(n)$ возвращает восьмеричное (octal) представление значения десятичного числа n в виде строки;
- функция $Hex(n)$ возвращает шестнадцатеричное (hexadecimal) представление значения десятичного числа n в виде строки;
- функция $ToBase(sn, k)$ преобразует десятичное значение числа n , заданное строкой sn , в строку, записанную в системе счисления с основанием k ($k = 2..36$) и возвращает эту строку;
- расширение $sn.ToBase(k)$ делает то же самое.

1. Перевод десятичного числа n в систему счисления по основанию k ($k = 2..36$)



Найти количество восьмерок в девятеричном представлении значения выражения $37^{59} - 19^{48}$

```
## uses School;  
(37bi ** 59 - 19bi ** 48).ToBase(9).CountOf('8').Print
```

А вот так выглядит это число в десятичном виде:

334 119 420 856 080 810 951 048 371 564 629 508 203 083 152 249 809
704 909 293 613 155 284 562 365 076 547 377 757 919 932

2. Перевод числа, заданного символьной строкой sn , представляющей значение в системе счисления по основанию $k = 2..36$ в десятичную систему счисления



- функция $Dec(sn, k)$ возвращает значение типа $int64$;
- функция $DecBig(sn, k)$ возвращает значение типа $BigInteger$.

2. Перевод числа, заданного символьной строкой sn , представляющей значение в системе счисления по основанию $k = 2..36$ в десятичную систему счисления



Вычислить десятичное значение числа, которое в шестнадцатиричной записи имеет вид $\underbrace{0F0F0F \dots 0F}_{64 \text{ символа}}$

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program with two lines of code: `1 ## uses School;` and `2 DecBig('0F'*32, 16).Print`. Below the code editor is an output window titled "Окно вывода" containing the decimal number `6811299366900952671974763824040465167839410862684739061144563765171360567055`. The status bar at the bottom indicates "Компиляция прошла успешно (2 строк)" and "Строка 2 Столбец 25".

```
1 ## uses School;
2 DecBig('0F'*32, 16).Print
```

Окно вывода

6811299366900952671974763824040465167839410862684739061144563765171360567055

Окно вывода | Список ошибок | Сообщения компилятора

Компиляция прошла успешно (2 строк) Строка 2 Столбец 25

3. Нахождение минимума и максимума последовательности целых чисел s за один проход



- функция $MinMax(s)$ возвращает кортеж (Min, Max) . Типом данных может быть *integer* или *int64*;
- расширение $s.MinMax$ делает то же самое.



3. Нахождение минимума и максимума последовательности целых чисел s за один проход

С клавиатуры вводится последовательность чисел, завершающаяся нулем (ноль в последовательность не входит). Найти и вывести минимальное и максимальное среди введенных значений

```
1  ##
2  uses School;
3  var (min, max) := ReadSeqIntegerWhile(t -> t <> 0).MinMax;
4  $'Минимальное введенное значение равно {min}, максимальное {max}'.Print
```

Окно вывода

```
27 152 -45 62 4 11 -23 0
Минимальное введенное значение равно -45, максимальное 152
```

Окно вывода | Список ошибок | Сообщения компилятора

Компиляция прошла успешно (3 строк) Строка 4 Столбец 7

4. Нахождение НОД и НОК пары чисел a и b



- функция $\text{НОД}(a, b)$ возвращает НОД типа *integer* или *int64*;
- расширение $t.\text{НОД}$ возвращает НОД для кортежа $t = (a, b)$ с данными типа *integer* или *int64*;
- функция $\text{НОК}(a, b)$ возвращает НОК типа *int64*;
- функция $\text{НОДНОК}(a, b)$ возвращает кортеж вида (НОД, НОК) для пары чисел a и b типа *int64*

Имеются англоязычные синонимы *GCD* для НОД и *LCM* для НОК

4. Нахождение НОД и НОК пары чисел a и b



Сократить дробь 360/2940

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program named 'Program 1.pas' with the following code:

```
1  ## uses School;  
2  var (a, b) := ReadInteger2;  
3  var d := НОД(a, b);  
4  $'{a}/{b} = {a div d}/{b div d}'.Print
```

The 'Окно вывода' (Output window) at the bottom shows the program's output:

```
360 2940  
360/2940 = 6/49
```

The status bar at the bottom indicates 'Компиляция прошла успешно (4 строк)' and 'Строка 4 Столбец 39'.

15

Конференция «Использование системы программирования PascalABC.NET
в обучении программированию», 13.11.2021

Южный федеральный университет
г. Ростов-на-Дону

5. Разложение числа n на простые множители. Результат помещается в список *List*



- функция *Factorize(n)* выполняет разложение числа n типа *integer* или *int64*;
- расширение *n.Factorize* делает то же самое

Тип элементов возвращаемого списка *List* совпадает с типом числа n

5. Разложение числа n на простые множители. Результат помещается в список *List*



Получить все четырехзначные натуральные числа, которые являются произведением пяти различных простых делителей. Числа упорядочить по возрастанию суммы их цифр, а при равной сумме – по возрастанию самих чисел.

```
1 uses School;
2
3 function Годится(n: integer): boolean;
4 begin
5     var L := n.Factorize;
6     Result := L.Count = 5 ? L.Distinct.Count = 5 : False
7 end;
8
9 begin
10    (1000..9999).Where(n -> Годится(n)).OrderBy(n -> n.Digits.Sum).ThenBy(n -> n).Print
11 end.
```

2310 2730 5610 6006 6510 7410 9030 3570 4290 4830 6090 6270 6630
8610 3990 7590 7770 9282 9570 7854 8970 9690 9870 8778

Компиляция прошла успешно (11 строк) Строка 6 Столбец 36

5. Разложение числа n на простые множители. Результат помещается в список *List*



Получить все четырехзначные натуральные числа, которые являются произведением пяти различных простых делителей. Числа упорядочить по возрастанию суммы их цифр, а при равной сумме – по возрастанию самих чисел.

```
1  uses School;
2
3  function Годится(n: integer): boolean;
4  begin
5      var L := n.Factorize;
6      Result := L.Count = 5 ? L.Distinct.Count = 5 : False
7  end;
8
9  begin
10     (1000..9999).Where(n -> Годится(n)).OrderBy(n -> n.Digits.Sum).ThenBy(n -> n).Print
11 end.
```

6. Нахождение простых чисел и проверка числа на простоту



- функция $Primes(n)$ возвращает список $List$, содержащий простые числа на отрезке $[2;n]$;
- функция $Primes(m, n)$ возвращает список $List$, содержащий простые числа на отрезке $[m;n]$;
- функция $FirstPrimes(n)$ возвращает список $List$, содержащий первые n простых чисел;
- функция $PrimeDivisorsCount(n)$ возвращает количество простых делителей числа n ;
- расширение $n.PrimeDivisorsCount$ делает то же самое;
- расширение $n.IsPrime$ возвращает $True$, если n – простое и $False$ в противном случае.

Тип переменных – *integer*, но $n.IsPrime$ также допускает тип *int64*

6. Нахождение простых чисел и проверка числа на простоту



На интервале [600 000 000;900 000 000] найти и вывести простые числа, сумма цифр которых равна 77.

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program named 'School211015.pas' with three lines of code:

```
1  ## uses School;  
2  Primes(600000000, 900000000).Where(n -> n.Digits.Sum = 77).Println;  
3  Print(Milliseconds/1000)
```

The output window below shows a grid of prime numbers. The number '6.122' is highlighted in yellow, indicating the execution time in seconds. The status bar at the bottom indicates 'Компиляция прошла успешно (3 строк)' and 'Строка 3 Столбец 25'.

6. Нахождение простых чисел и проверка числа на простоту



На интервале [600 000 000;900 000 000] найти и вывести простые числа, сумма цифр которых равна 77.

```
1  ## uses School;  
2  var o := new Stopwatch;  
3  o.Start;  
4  var a := Primes(600000000, 900000000).Where(n -> n.Digits.Sum = 77).ToArray;  
5  o.Stop;  
6  a.Println;  
7  Print(o.Elapsed)
```

A screenshot of a Windows console window titled "Program1.exe". The window displays a list of prime numbers whose digits sum to 77, arranged in two rows. The first row contains: 699899999, 779999999, 788999999, 789999989, 797999999, 798899999, 799898999, 799899899, 799999799, 879999899, 888989999, 898799999. The second row contains: 898898999, 898979999, 898988999, 898989899, 898999979, 899879999, 899889899, 899898899, 899978999, 899988899, 899989889. Below the list, the text "00:00:03.2416839 Программа завершена, нажмите любую клавишу . . ." is visible, indicating the execution time and completion of the program.

```
Program1.exe  
699899999 779999999 788999999 789999989 797999999 798899999 799898999 799899899 799999799 879999899 888989999 898799999  
898898999 898979999 898988999 898989899 898999979 899879999 899889899 899898899 899978999 899988899 899989889  
00:00:03.2416839 Программа завершена, нажмите любую клавишу . . .
```

7. Получение всех цифр в записи натурального числа n в порядке их следования



- функция *Digits(n)* возвращает список *List<integer>*, содержащий цифры числа n типа *integer* или *int64*;
- расширение *n.Digits* делает то же самое.

7. Получение списка *List*, содержащего все цифры числа n в порядке их следования



Найти сумму всех цифр числа 20!

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program named 'School211015.pas' with two lines of code: '1 ## uses School;' and '2 (1..20).Product.Digits.Sum.Print'. Below the code editor is an 'Окно вывода' (Output window) displaying the number '54'. The status bar at the bottom indicates 'Компиляция прошла успешно (2 строк)' and 'Строка 2 Столбец 9'.

```
1 ## uses School;
2 (1..20).Product.Digits.Sum.Print
```

Окно вывода

54

Окно вывода | Список ошибок | Сообщения компилятора

Компиляция прошла успешно (2 строк) | Строка 2 Столбец 9

8. Все делители натурального числа n , включая 1 и n



- функция $Divisors(n)$ для числа n типа $integer$ возвращает список всех его делителей $List<integer>$ в порядке возрастания;
- расширение $n.Divisors$ делает то же самое;
- функция $DivisorsCount(n)$ возвращает количество делителей числа n типа $integer$;
- расширение $n.DivisorsCount$ делает то же самое;

8. Все делители натурального числа n , включая 1 и n



Посчитать количество чисел из интервала [5243;13263], имеющих ровно 10 делителей.

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program with two lines of code:

```
1  ## uses School;  
2  (5243..13263).Where(n -> n.Divisors.Count = 10).Count.Print
```

The output window at the bottom displays the number 104. The status bar at the bottom indicates "Компиляция прошла успешно (2 строк)" and "Строка 1 Столбец 4".

25

Конференция «Использование системы программирования PascalABC.NET в обучении программированию», 13.11.2021

Южный федеральный университет
г. Ростов-на-Дону

9. Вычисление тригонометрических функций вещественного аргумента x , заданного в градусной мере



- функция *SinDegrees*(x) возвращает значение $\sin(x)$;
- функция *CosDegrees*(x) возвращает значение $\cos(x)$;
- функция *TanDegrees*(x) возвращает значение $\text{tg}(x)$

Аргумент x задается в градусной мере.

```
## uses School;  
$'sin(30°) = SinDegrees(30)'.Println; // sin(30°) = 0.5  
$'cos(0°) = CosDegrees(0)'.Print // cos(0°) = 1
```

10. Генерация наборов случайных вещественных чисел с заданным количеством знаков в дробной части



- функция $ArrRandomReal(n, a, b, t)$ возвращает массив длины n , заполненный случайными вещественными числами из промежутка $[a;b)$ с t знаками в дробной части;
- функция $SeqRandomReal(n, a, b, t)$ делает то же самое для для последовательности;
- функция $MatrRandomReal(m, n, a, b, t)$ возвращает массив размером $m \times n$, заполненный случайными вещественными числами из промежутка $[a;b)$ с t знаками в дробной части.



10. Генерация наборов случайных вещественных чисел с заданным количеством знаков в дробной части

Найти суммы по каждой строке вещественного массива размером 6×9 , заполненного числами из диапазона $[-99;99]$, имеющими один знак в дробной части

```
PascalABC.NET 3.8.2
Файл Правка Вид Программа Сервис Модули Помощь
School211015.pas *Program1.pas*
Глобальные имена
1  ## uses School;
2  var a := MatrRandomReal(6, 9, -99, 99, 1);
3  a.Println(6,1);
4  Writeln;
5  a.Rows.Select(r -> r.Sum).Print
Окно вывода
70.7 -69.5 -74.3 -62.4 -38.7 -9.1 -10.1 -88.4 45.9
8.7 45.2 59.8 -77.8 -15.0 63.1 -59.1 8.1 -62.5
37.3 -89.9 28.1 66.5 14.7 14.7 77.3 49.3 67.7
20.1 58.2 63.0 -59.5 -51.8 70.7 37.3 -51.9 -47.7
-20.8 -25.7 -29.2 59.4 67.6 68.7 -69.9 6.7 83.3
-52.4 -61.7 29.5 21.2 -2.2 -82.8 92.7 76.2 -50.6

-235.9 -29.5 265.7 38.4 140.1 -30.1
Окно вывода Список ошибок Сообщения компилятора
Компиляция прошла успешно (5 строк) Строка 2 Столбец 9
```

Конференция «Использование системы программирования PascalABC.NET в обучении программированию», 13.11.2021

Южный федеральный университет
г. Ростов-на-Дону



10. Генерация наборов случайных вещественных чисел с заданным количеством знаков в дробной части

Найти суммы по каждой строке вещественного массива размером 6×9 , заполненного числами из диапазона $[-99;99]$, имеющими один знак в дробной части

```
1  ## uses School;
2  var a := MatrRandomReal(6, 9, -99, 99, 1);
3  a.Println(6,1);
4  Writeln;
5  a.Rows.Select(r -> r.Sum).Print
```

11. Построение таблиц истинности



Функция *TrueTable*((*a*, *b*) -> *f*(*a*, *b*)) возвращает матрицу типа *boolean*, содержащую таблицу истинности для заданной лямбда-функции двух аргументов;

Имеются разновидности функции, в которых можно указать от двух до пяти аргументов.

Полученную матрицу можно вывести в наглядном виде, используя процедуру *TrueTablePrint* с различным набором аргументов.

11. Построение таблиц истинности



Процедура `TrueTablePrint(a)` выводит таблицу истинности на основе матрицы `a`, полученной посредством функции `TrueTable`;

Процедура `TrueTablePrint(a, f)` при `f = 0` выводит только строки, для которых первичная логическая функция имеет значение `False`, при `f = 1` – только строки, в которых оно имеет значение `True`, для иных значений `f` выводятся все строки.

Процедура `TrueTablePrint(a, f, s)` может иметь третий параметр `s` типа `string`, по умолчанию равный `'abcd'`. Этот параметр позволяет задать символы, которые будут именовать колонки таблицы истинности.

11. Построение таблиц истинности



Построить таблицу истинности для функции трех переменных

$$F(x, y, z) = x \rightarrow (y \vee \neg y \wedge \neg z)$$

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program with three lines of code: 1. `## uses School;`, 2. `var a := TrueTable((x, y, z) -> x <= (y or not y and not z));`, and 3. `TrueTablePrint(a, 2, 'xyz')`. Below the code is an output window titled "Окно вывода" which displays a truth table for the function F(x, y, z). The table has four columns: x, y, z, and F. The first row is a header with dashes under each column. The following eight rows list all possible combinations of x, y, and z (0 or 1) and the corresponding value of F. The status bar at the bottom indicates "Компиляция прошла успешно (3 строк)" and "Строка 1 Столбец 16".

```
1  ## uses School;
2  var a := TrueTable((x, y, z) -> x <= (y or not y and not z));
3  TrueTablePrint(a, 2, 'xyz')
```

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

12. Замены символов и подстрок в строке



- процедура *ReplaceLast(s, a, b)* заменяет в строке *s* последнее вхождение подстроки *a* подстрокой *b*;
- функция *ReplaceMany(s, source, target)* возвращает строку, в которой для исходной строки *s* все вхождения подстрок или символов из *source* заменяются на подстроки или символы из *target*. При замене подстрок параметры *source* и *target* должны быть массивами или списками;
- расширение *s.ReplaceMany(source, target)* делает то же самое;
- процедура *SwapSubstr(s, ss1, ss2)* обменивает местами все подстроки *ss1* и *ss2* в строке *s*;
- расширение *s.SwapSubstr(ss1, ss2)* делает то же самое.

12. Замены символов и подстрок в строке



В заданной строке заменить все символы '+' на '-', '/' на '*', а символы '=' удалить. Далее заменить все подстроки 'ac' на 'X', а 'com' на 'RU'.

A screenshot of the PascalABC.NET 3.8.2 IDE. The main window shows a Pascal program with the following code:

```
2 var s := ReadString;  
3 s := s.ReplaceMany('+/=', '-*').ReplaceMany(|'ac', 'com'|, |'X', 'RU'|);  
4 s.Print
```

The output window below shows the result of running the program:

```
http:www.x2ac0ez.com, acmpre.com 25+7/2== 11Ac  
http:www.x2X0ez.RU, Xmpre.RU 25-7*2 11Ac
```

The status bar at the bottom indicates "Компиляция прошла успешно (4 строк)" and "Строка 4 Столбец 8".

Дополнительные подпрограммы и расширения



- для типа *string* без подключения библиотеки **School** расширения *Combinations*, *Permutations* и *Cartesian* возвращают последовательность массивов символов. При подключенной библиотеке они возвращают последовательность строк;
- значение типа любой переменной *o* или выражения в терминах **PascalABC.NET** можно вывести посредством процедур *PrintType(o)* и *PrintTypeLn(o)*.

Кое-что о реализации библиотеки School



LPrimes: List<integer> – внутренний список простых чисел на отрезке [2 ; 46 349], содержащий 4 792 элемента. Строится при инициализации School посредством модифицированного решета Эратосфена.

$$46\ 349^2 = 2\ 148\ 229\ 801$$

$$\text{MaxInt} = 2\ 147\ 483\ 647 = 2^{31}-1$$

$$46\ 337^2 = 2\ 147\ 117\ 569$$

366 078

Если хотя бы одна из границ отрезка, в котором ищутся простые числа, находится на числовом луче правее значения 46 349, используется сегментированное решето Эратосфена.



Благодарю за внимание!



37

**Конференция «Использование системы программирования PascalABC.NET
в обучении программированию», 13.11.2021**

**Южный федеральный университет
г. Ростов-на-Дону**