



# PascalABC.NET 2021

Михалкович С.С.

*Южный федеральный университет,*
*Институт математики, механики и компьютерных наук*

miks@sfedu.ru

Доклад на третьей Всероссийской научно-методической конференции
«Использование системы программирования PascalABC.NET
в обучении программированию»
(Ростов-на-Дону, 12-13 ноября 2021 г.)

# PascalABC.NET – история развития

Основная причина создания – обучение современному программированию

### 2007 год

Выпуск первой версии PascalABC.NET. Развитие методики преподавания PascalABC.NET: полигон – **детская** компьютерная школа мехмата ЮФУ (выпуск – 700 учащихся в год)

### 2015 год

Переход на свободную лицензию LGPL. Начало активного использования PascalABC.NET в школах России и на олимпиадах по программированию

### **2015**-2021 гг.

Активное развитие языка PascalABC.NET. Использование **лучших идей современных языков программирования**. Приближение по краткости записи к языку Python.

### 2017-2021 гг.

Развитие графических модулей для обучения (2D и 3D графика), развитие модуля School для решения базовых задач информатики

### 2021 год

Компьютерный ЕГЭ - усложнение задач. **Третья** Всероссийская конференция по использованию PascalABC.NET в учебном процессе

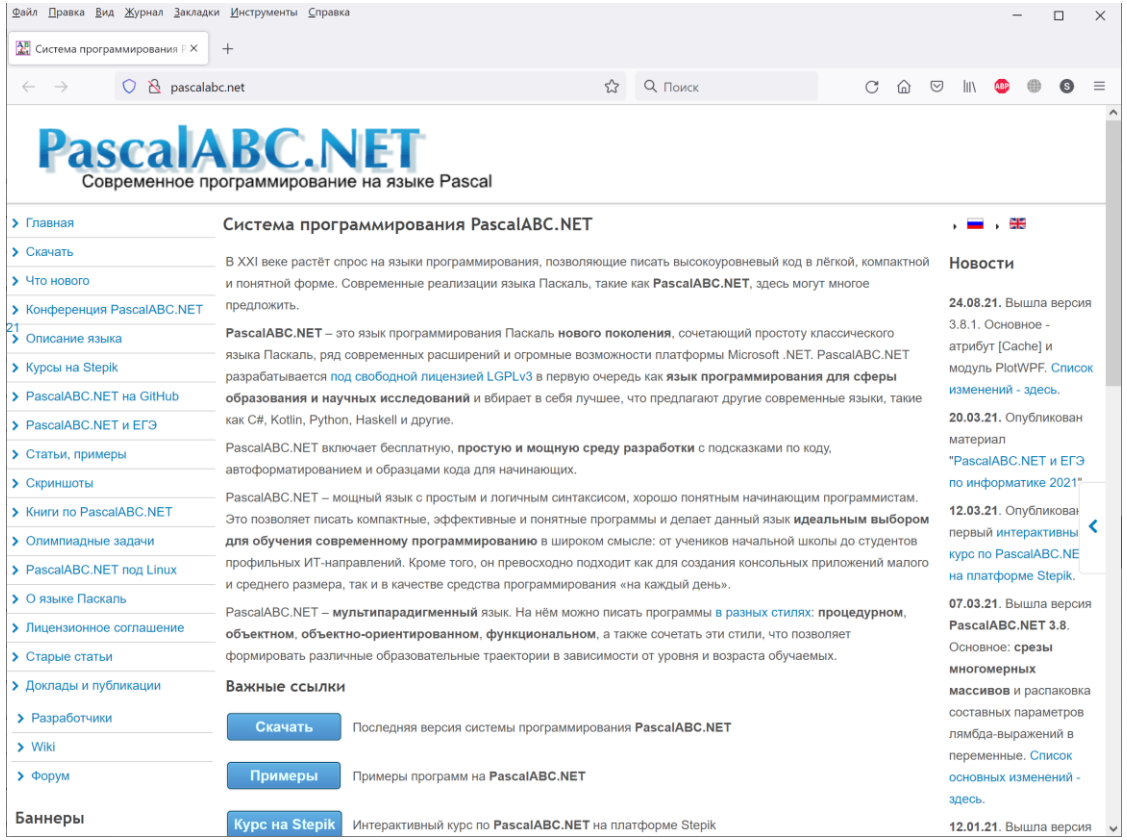
**Слайд конференции 2020 г.**

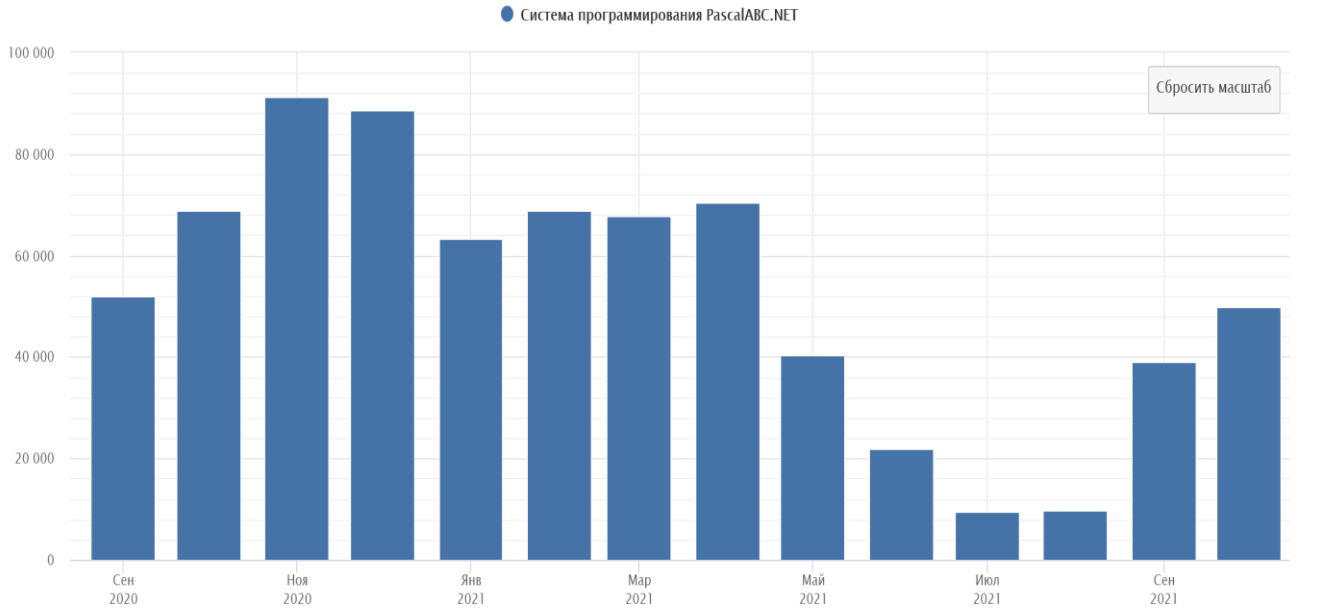
# Области использования PascalABC.NET

- Для обучения **школьников средних и старших классов** современному программированию в **Компьютерной школе мехмата ЮФУ** (выпуск **700** учащихся в год, огромный опыт работы – с 1995 года). Уникальные методики обучения, диктующие изменения в языке PascalABC.NET и его библиотеках
- Для обучения **младших школьников**
- В **олимпиадах** по программированию
- При **подготовке к ЕГЭ** по информатике
- Для обучения **студентов ИТ направлений** (Направление Фундаментальная информатика на мехмате ЮФУ, набор 110 человек)
- Как средство программирования «**на каждый день**»


**Слайд конференции 2020 г.**

# PascalABC.NET – распространение за пределами ЮФУ




| ГЕОГРАФИЯ | ПОСЕТИТЕЛИ | | А/I, % |
|---|---|---|---|
| Россия | 34,856 | 69.79% | 94.17 |
| Беларусь | 9,825 | 19.67% | 636.57 |
| Республика Молдова | 1,888 | 3.78% | 718.95 |
| Казахстан | 825 | 1.65% | 34.54 |
| Украина | 778 | 1.56% | 128.97 |
| Узбекистан | 403 | 0.81% | 106.22 |
| Киргизия | 273 | 0.55% | 122.48 |
| Туркменистан | 249 | 0.50% | 1,201.66 |
| Азербайджан | 160 | 0.32% | 76.94 |
| Соединенные Штаты Америки | 87 | 0.17% | 6.59 |

- Посещаемость сайта – **50000** посетителей в месяц
- Пользователи из Беларуси, Молдовы, Украины, Казахстана, США

# Количество скачиваний PascalABC.NET по данным официального сайта

2021 год: **5 миллионов скачиваний** с начала проекта, **2000** скачиваний в день

## Статистика скачиваний

**PABC.NET по дням:**

| Дата | Количество |
|---|---|
| 2021-10-29 | 1316 |
| 2021-10-28 | 2294 |
| 2021-10-27 | 2503 |
| 2021-10-26 | 2116 |
| 2021-10-25 | 1896 |
| 2021-10-24 | 1314 |
| 2021-10-23 | 1226 |
| 2021-10-22 | 1578 |
| 2021-10-21 | 1950 |
| 2021-10-20 | 1727 |
| 2021-10-19 | 1692 |
| 2021-10-18 | 1676 |
| 2021-10-17 | 1377 |

**PABCFull.NET по дням:**

| Дата | Количество |
|---|---|
| 2021-10-29 | 20 |
| 2021-10-28 | 35 |
| 2021-10-27 | 29 |
| 2021-10-26 | 37 |
| 2021-10-25 | 24 |
| 2021-10-24 | 34 |
| 2021-10-23 | 24 |
| 2021-10-22 | 31 |
| 2021-10-21 | 30 |
| 2021-10-20 | 30 |
| 2021-10-19 | 36 |
| 2021-10-18 | 34 |
| 2021-10-17 | 37 |

**PABCMini.NET по дням:**

| Дата | Количество |
|---|---|
| 2021-10-29 | 225 |
| 2021-10-28 | 294 |
| 2021-10-27 | 382 |
| 2021-10-26 | 313 |
| 2021-10-25 | 259 |
| 2021-10-24 | 230 |
| 2021-10-23 | 229 |
| 2021-10-22 | 276 |
| 2021-10-21 | 328 |
| 2021-10-20 | 308 |
| 2021-10-19 | 274 |
| 2021-10-18 | 323 |
| 2021-10-17 | 212 |

# Как учителю изучить новые возможности PascalABC.NET?

# Прежде всего – понять необходимость

- Стандартный Паскаль (его коды можно увидеть на ЕГЭ) сильно устарел
- Использование стандартного Паскаля вызывает отторжение учеников потому что современные языки предлагают кардинально новые возможности
- PascalABC.NET создан именно с целями:
  - как язык Паскаль нового поколения: эффективный, понятный, лаконичный
  - как язык, берущий от современных языков программирования лучшее: строгую типизацию от Java, C#, C++, компактность записи – от Python
  - как язык, нацеленный прежде всего на сферу образования и процесс обучения
- PascalABC.NET активно развивается вместе с промышленными языками программирования, берёт от них лучшее для сферы современного образования

# Вредные мысли

1. Надо учить базовому Паскалю – это стандарт (точно **НЕТ**. Стандарт старый и его никто не использует)
2. Надо переходить на Питон – он везде и простой (мысль хорошая, тогда вам – на другие конференции ☺, их **НЕТ** ☺). А если серьёзно – **не может быть единого языка для обучения**, и Питон – старый язык (ему 30 лет), имеющий свои «родовые травмы», ухудшающие обучение
3. А перейду-ка я на C++ – он эффективен и популярен на олимпиадах! (**НЕТ**, C++ – сложнейший язык, сравнительно редко используемый в ИТ-индустрии, с непростой стандартной библиотекой)
4. В PascalABC.NET так много новых возможностей – школьникам это не нужно! (**НЕТ.**
   **Во-первых**, далеко не все возможности нужно использовать в обучении.
   **Во-вторых**, эффективная сдача ЕГЭ обязательно требует язык с современными возможностями и библиотеками)
5. Научу я новым возможностям, а на ЕГЭ или ещё где-то их не окажется (**НЕТ**: на ЕГЭ практически повсеместно использовалась версия PascalABC.NET не ниже 3.7.1 – этого достаточно, за этим надо проследить; на распространенных олимпиадных сайтах – тоже последние версии PascalABC.NET)

# Необходимый минимум возможностей PascalABC.NET

(базовый уровень)

# Необходимый минимум

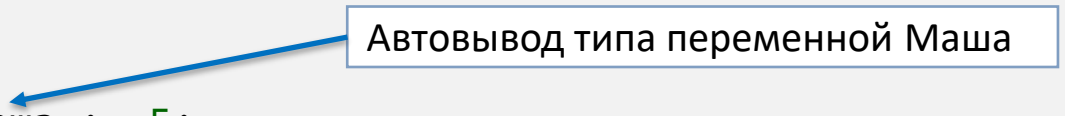
- Описание переменных внутри блока
- Автоматический вывод типов
- Русские имена переменных
- Процедура Print – для вывода данных
- Функции ReadInteger, ReadReal, ReadString – для ввода данных
- Операции += *=
- Стандартные функции Swap(a,b), Min(x,y), Max(x,y)
- Множественное присваивание, множественная инициализация
- Операция возведения в степень **
- Операция x in 1..10 проверки принадлежности к диапазону
- Цикл loop
- Цикл for с описанием переменной цикла в заголовке
- Тип длинного целого BigInteger
- Командный режим записи программ ##

# Автоматический вывод типов

Переменная должна объявляться и инициализироваться как можно ближе к моменту своего первого использования. Компилятор выведет тип автоматически.

```
{ Тема. Переменные

  Задание. Маша съела 5 конфет.
    Витя съел на 2 конфеты больше.
    Сколько всего конфет съели Маша и Витя?
}
begin
  var Маша := 5;
  var Витя := Маша + 2;
  Println(Витя + Маша);
end.
```



# Описание переменной с вводом

Переменную можно инициализировать при описании вводом данных

```
{ Тема. Переменные

  Задание. Дано количество конфет, которое съела Маша.
    Витя съел на 2 конфеты больше.
    Сколько всего конфет съели Маша и Витя?
}
begin
  var Маша := ReadInteger('Сколько конфет съела Маша:');
  var Витя := Маша + 2;
  Println(Витя + Маша);
end.
```

# Ввод нескольких. Процедура Swap

ReadInteger2 – для ввода двух целых,
Swap(a,b) – поменять местами значения a и b

```
{ Тема. Переменные

  Задание. Дано количество конфет, которое было у Маши, Вити и Пети.
    Вначале Маша и Витя поменялись своими конфетами.
    Затем Витя поменялся своими конфетами с Петей.
    Сколько конфет стало у Маши, Вити и Пети?
}
begin
  var (Маша, Витя, Петя) := ReadInteger3;
  Swap(Маша, Витя);
  Swap(Витя, Петя);
  Println(Маша, Витя, Петя);
  Println(Min(Маша, Витя));
  Println(Max(Маша, Витя, Петя));
end.
```

```
Swap(x,y):
var t := x;
x := y;
y := t;
```

```
Min(x,y):
var min: integer;
if x<y then
  min := x
else min := y;
```

# Цикл loop, Print, ReadInteger, *=, +=

Цикл loop используется в случаях, когда номер повторения цикла не важен

**Слайд конференции 2020 г.**

## Пример 1. Степени двойки

```
begin
  var n := 11;
  var a := 1;
  loop n do
  begin
    Print(a); // данные разделяются пробелами
    a *= 2
  end;
end.
```

```
1 2 4 8 16 32 64 128 256 512 1024
```

## Пример 2. Сумма чисел, попадающих в диапазон 10..20, среди n введённых

```
begin
  var n := ReadInteger('Введите n:');
  var sum := 0;
  loop n do
  begin
    var x := ReadInteger;
    if x in 10..20 then
      sum += x;
  end;
  Print('Сумма чисел в диапазоне 10..20 =',sum);
end.
```

Эффективное использование внутриблочных переменных

```
2475
```

# Длинные целые BigInteger

Тип длинного целого BigInteger используется, когда нам необходимо использовать в программе большие целые значения

## Вычисление $2^{1000}$

```
begin
  var n := 1000;
  var p: BigInteger := 1;
  loop n do
    p *= 2;
  Print(p);
end.
```

```
1071508607186267320948425049060001810561404
8117055336074437503883703510511249361224931
9837881569585812759467291755314682518714528
5692314043598457757469857480393456777482423
0985421074605062371141877954182153046474983
5819412673987675591655439460770629145711964
7768654216766042983165262438683720566806937
6
```

## Сколько двоек в записи числа $7^{60} + 49^{29}$

```
begin
  var x := 7bi ** 60 + 49bi ** 29;
  var count := 0;
  while x>0 do
  begin
    if x mod 10 = 2 then
      count += 1;
    x := x div 10;
  end;
  Print(count);
end.
```

Операция возведения в степень

```
6
```

# Цикл for и описание переменной в заголовке цикла

Параметр цикла for должен описываться в заголовке цикла. Тогда его невозможно будет использовать для других целей. Одна переменная – для одной цели.

```
begin
  for var i:=1 to 9 do
    Print(i);
  Println; // Здесь переменная i недоступна
  for var i:=1 to 9 do
    Print(2*i-1);
end.
```

```
1 2 3 4 5 6 7 8 9
1 3 5 7 9 11 13 15 17
```

# Командный режим записи программ

Командный режим записи программ начинается с ## и позволяет опускать внешний begin-end. Он служит для коротких программ в несколько строк и используется обычно в демонстрационных целях и в Jupiter-ноутбуках PascalABC.NET

```
##
var (a,b) := ReadInteger2;
Print(a + b, a * b);
```

```
##
function SumSquares(a,b: integer) := a*a + b*b;

Print(SumSquares(3,4));
```

```
##
Print(2bi ** 100);
```

```
##
var a := ArrRandomInteger(10);
a.Println;
a.Where(x -> x.Divs(3)).Println;
```

```
##
uses Graph3D;
Sphere(0,0,0,2);
```

```
##
uses GraphWPF;

OnMouseDown := (x,y,mb) -> Circle(x,y,5);
```

```
##
uses School;
var x := 123456;
bin(x).Println;
```

# Возможности PascalABC.NET для обучения. **Уровень олимпиадных задач и профильного ЕГЭ**

# Возможности языка для обучения. Уровень ЕГЭ и олимпиадных задач

- Строки, методы строк, операции со строками
- Динамические массивы, методы массивов
- Срезы массивов, операции с массивами
- Лямбда-выражения
- Списки
- Множества
- Функции, рекурсия
- Основы работы с файлами (ЕГЭ, олимпиады)

Великолепная презентация К.Полякова «Новые возможности PascalABC.NET»

# Цикл foreach по символам строки

Чтобы перебрать все символы в строке, проще использовать цикл foreach

Количество букв и цифр в строке

```
begin
  var s := 'ABC 123';
  var count := 0;
  foreach var c in s do
    if c.IsLetter or c.IsDigit then
      count += 1;
  Print(count)
end.
```

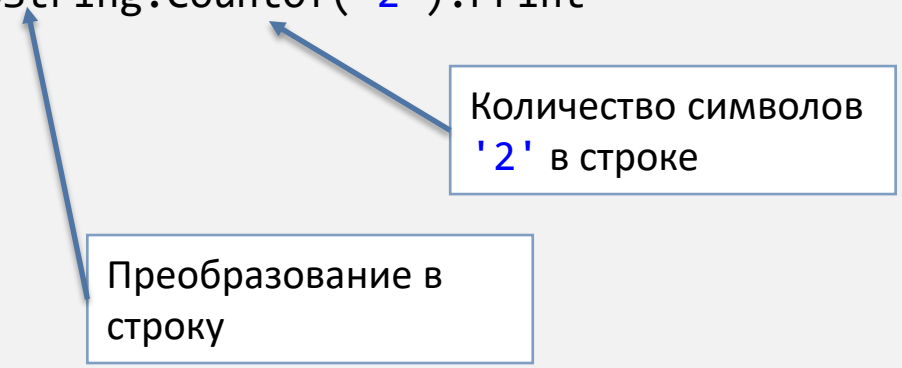
foreach – для каждого символа c в строке s

Методы IsLetter и IsDigit для символов

# Преобразование числа в строку и метод CountOf

Сколько двоек в записи числа $7^{60} + 49^{29}$

```
##
var x := 7bi ** 60 + 49bi ** 29;
x.ToString.CountOf('2').Print
```




```
6
```

# Динамические массивы

Динамические массивы array of T имеют огромное количество стандартных методов для решения часто встречающихся задач.

```
##
var a := |1, 3, 5, 7, 9|;
foreach var x in a do
  Print(x);
Println;
a := Arr(1..7);
a.Println;
for var i := 0 to a.Length - 1 do
  a[i] += 1;
a.Println;
```

```
1 3 5 7 9
1 2 3 4 5 6 7
2 3 4 5 6 7 8
```

```
##
var a := ArrRandomInteger(10,1,9);
a.Println;
a[:5].Println;
a[5:].Println;
a[::-1].Println;
Println(a[^1],a[^2]);
Println(a.Min, a.Max, a.Sum, a.Average);
a.Where(x->x mod 2 = 0).Println;
a.Count(x->x mod 2 = 0).Println;
a.Order.Println;
```

```
8 8 2 3 1 6 2 5 4 4
8 8 2 3 1
6 2 5 4 4
4 4 5 2 6 1 3 2 8 8
4 4
1 8 43 4.3
8 8 2 6 2 4 4
7
1 2 2 3 4 4 5 6 8 8
```

# Цепочка методов

Фильтрация, преобразование и сортировка элементов

**Слайд конференции 2020 г.**

## Цепочка методов

```
var a := ArrRandomInteger(10);
a.Println;
        Лямбда-условие            лямбда-преобразование
a.Where(x -> x mod 2 = 0).Select(x -> x*2).Order.Println;
```

```
67 76 19 93 5 72 58 22 83 4
8 44 116 144 152
```

Как англичане воспринимают этот код:

```
массив.Выбрать(x такие что x mod 2 = 0).Преобразовать(x в x*2).Отсортировать.Вывести;
```

Восприятие естественное, эффективность почти та же

# Списки

Списки List – это динамические массивы с возможностью добавления в конец

Определение всех нетривиальных делителей числа

```
begin
  var lst := new List<integer>;
  var N := 12345;
  for var i:=2 to N-1 do
    if N mod i = 0 then
      lst.Add(i);
  lst.Println;
end.
```

```
3 5 15 823 2469 4115
```

# Функции, рекурсия

В ЕГЭ есть задачи на использование функций и на рекурсию

## Определение короткой функции

```
function SumSq(a,b: integer) := a*a + b*b;

begin
  Print(SumSq(3,4))
end.
```

## Страшная рекурсия из ЕГЭ

```
function F(n: integer): integer;
begin
  if n = 1 then
    Result := 1
  else if n.IsEven then
    Result := n + F(n - 1)
  else Result := 2 * F(n - 2);
end;

begin
  F(26).Print
end.
```

# Основы работы с файлами

Для считывания из файла достаточно связать поток ввода с файлом

## Считать строку

```
begin
  Assign(input,'24.txt');
  var s := ReadString;
  ...
end.
```

## Считать массив из N целых

```
begin
  Assign(input,'26.txt');
  var N := ReadInteger;
  var a := ReadArrInteger(N);
  ...
end.
```

# Нововведения, появившиеся в PascalABC.NET в 2020 – 21 гг.

# Новое за 2020-2021 гг.

- Литералы BigInteger
- uses и объявления подпрограмм в коротких программах
- Срезы многомерных массивов
- Атрибут [Cache]
- not in
- Размещения и размещения с повторениями
- Методы расширения строк s.IsInteger, s.IsReal
- CountOf для последовательностей
- Sort с проекцией на ключ
- EachCount для последовательностей
- PartialSum для числовых последовательностей
- Стандартные модули PlotWPF, Turtle
- Стандартный модуль XLSX (Богданов А.А.)
- Нововведения в модуле School (Осипов А.В.)

# Литералы BigInteger

Чтобы использовать целую константу типа BigInteger, достаточно использовать суффикс bi после целочисленной константы

Вычисление $2^{1000}$

```
begin
  var n := 1000;
  var p := 1bi;
  loop n do
    p *= 2;
  Print(p);
end.
```

```
1071508607186267320948425049060001810561404
8117055336074437503883703510511249361224931
9837881569585812759467291755314682518714528
5692314043598457757469857480393456777482423
0985421074605062371141877954182153046474983
5819412673987675591655439460770629145711964
7768654216766042983165262438683720566806937
6
```

Сколько двоек в записи числа $7^{60} + 49^{29}$

```
##
var x := 7bi ** 60 + 49bi ** 29;
x.ToString.CountOf('2').Print
```

```
6
```

# Секция uses и объявления подпрограмм в командном режиме записи программ

В командном режиме ## теперь разрешены подключения внешних модулей с помощью uses и объявления подпрограмм

## Использование uses

```
##
uses Graph3D;
Sphere(0,0,0,2);
```

```
##
uses School;
var x := 123456;
bin(x).Println;
```

## Описание функций

```
##
function Divisors(n: integer): List<integer>;
begin
  Result := new List<integer>;
  for var i:=2 to n-1 do
  if n mod i = 0 then
    Result.Add(i);
end;

Divisors(2021).Println;
```

```
43 47
```

# Срезы многомерных массивов

В многомерных массивах также появились срезы:

```
##
var m := Matr(|1,2,3,4|,|5,6,7,8|,|9,10,11,12|);
m[:,:].Println;
m[1:,1:3].Println; // матрица - срез
m[1,:].Println;    // срез – строка
m[:,^1].Println;   // срез – столбец
```

```
   1   2   3   4
   5   6   7   8
   9  10  11  12
   6   7
  10  11
5 6 7 8
4 8 12
```

# Атрибут [Cache]

Атрибут [Cache] предназначен для кеширования результатов функции

```
##
// С кешированием
[Cache]
function fib(n: integer): integer :=
  if n in 1..2 then 1
  else fib(n-1) + fib(n-2);

// Без кеширования
function fib1(n: integer): integer :=
  if n in 1..2 then 1
  else fib1(n-1) + fib1(n-2);

Println(fib(42),MillisecondsDelta/1000);
Println(fib1(42),MillisecondsDelta/1000);
```

```
267914296 0.011
267914296 9.473
```

# Операция not in

Операция not in противоположна операции in:

```
##
var a := Arr(1,3,5);
if 4 not in a then
  Println('отсутствует');

var b := ArrRandomInteger(10,1,9);
b.Println;
b.Where(x -> x not in a).Println;
```

```
отсутствует
1 4 5 6 4 5 1 3 9 6
4 6 4 9 6
```

# Размещения и размещения с повторениями

Размещения a.Permutations(m) и размещения с повторениями a.Cartesian(m) дополняют методы

a.Combinations(m) – сочетания из n элементов по m

a.Permutations – перестановки из n элементов

```
##
var a := Arr(1,3,5);
a.Permutations(2).Println; // размещения из 3 элементов по 2
a.Cartesian(2).Println;    // размещения с повторениями из 3 элементов по 2
```

```
[1,3] [3,1] [1,5] [5,1] [3,5] [5,3]
[1,1] [1,3] [1,5] [3,1] [3,3] [3,5] [5,1] [5,3] [5,5]
```

# Методы расширения строк s.IsInteger и s.IsReal

Позволяют проверять, находится ли в строке целое или вещественное

```
##
var s := '123.4 3 5 6.6 a v 67';
var (si,sr) := (0,0.0);
foreach var w in s.ToWords do
  if w.IsInteger then
    si += w.ToInteger
  else if w.IsReal then
    sr += w.ToReal;
Print(si,sr);
```

```
75 130
```

# seq.CountOf(x) для последовательностей

seq.CountOf(x) позволяет найти в последовательности количество элементов, равных x. Это – сокращение для seq.Count(y -> y = x)

```
##
var a := Arr(1,3,5,7,1,2,1,3,1,5);
a.CountOf(1).Print;

var s := 'абракадабра';
s.CountOf('a').Print;
```

```
4 5
```

# Sort с проекцией на ключ

У стандартной процедуры Sort для сортировки составных данных появилась возможность указывать параметр – проекцию на ключ сортировки.

```
##
var pp := |('Сидоров',20),('Петров',22), ('Попов',20),('Иванов',22)|;
Sort(pp, p->p[1]); // Сортировка по возрасту
pp.Println;
Sort(pp, p->(p[1],p[0])); // Сортировка по возрасту, затем по фамилии
pp.Println;
```

```
(Сидоров,20) (Попов,20) (Петров,22) (Иванов,22)
(Попов,20) (Сидоров,20) (Иванов,22) (Петров,22)
```

# EachCount для последовательностей

Для последовательностей можно непосредственно вызывать метод EachCount, возвращающей частотный словарь элементов

```
##
var s := 'каракатица так и катится';
s.EachCount.Println;
var s1 := 'кто что где как что где что';
s1.ToWords.EachCount.PrintLines(x -> x.Key + ': '+ x.Value);
```

```
(к,4) (а,6) (р,1) (т,4) (и,3) (ц,1) ( ,3) (с,1) (я,1)
кто: 1
что: 3
где: 2
как: 1
```

# PartialSum для числовых последовательностей

Для числовых последовательностей определен метод PartialSum, возвращающий последовательность частичных сумм. PartialSum может быть применен в задании ЕГЭ 27

```
##
var a := Arr(1..7);
a.Println;
a.PartialSum.Println;
```

```
1 2 3 4 5 6 7
1 3 6 10 15 21 28
```

# Стандартный модуль PlotWPF

Стандартный модуль PlotWPF позволяет строить графики функций с различными параметрами

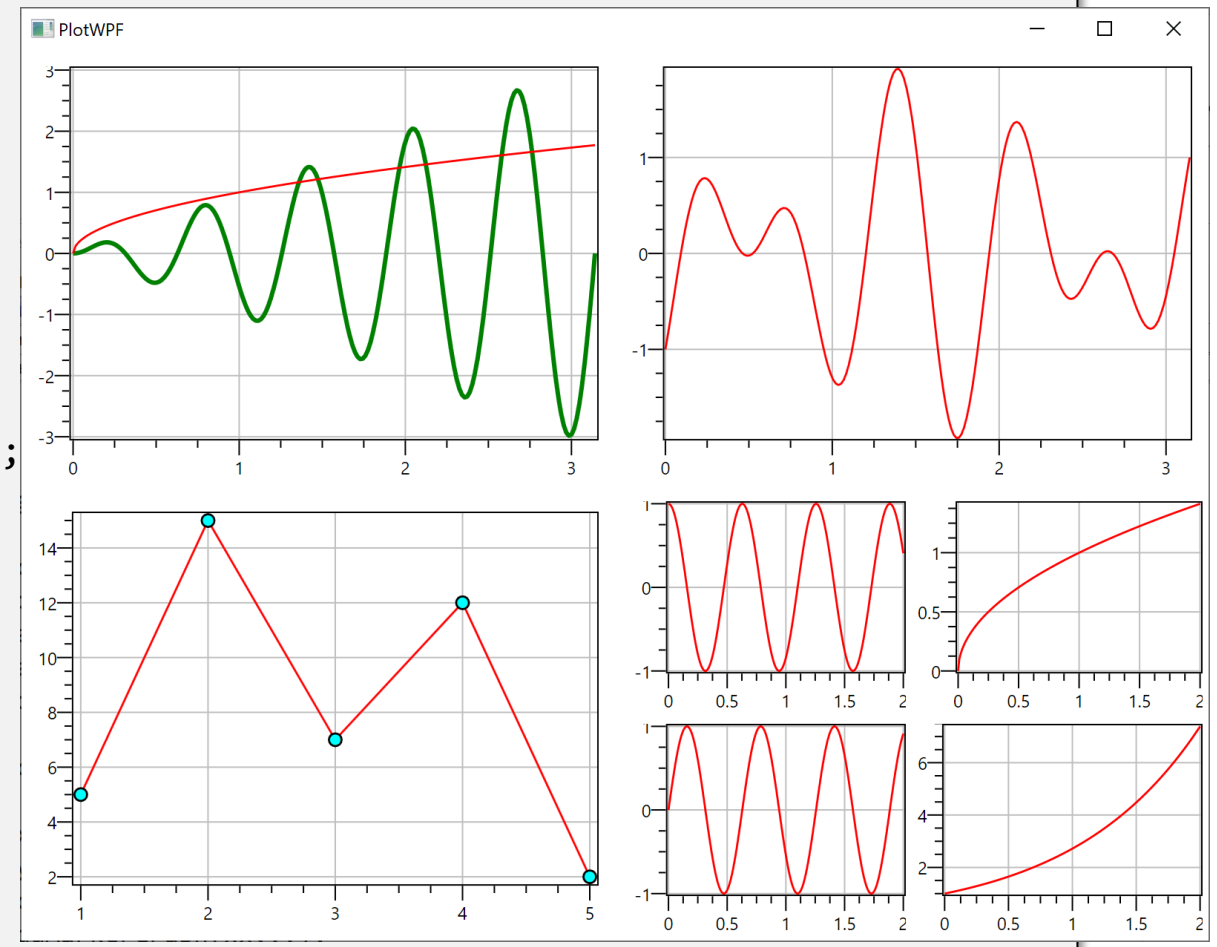
```
##
uses PlotWPF;

var g := new GridWPF(2,2,10);

var c := new LineGraphWPF(0,Pi,v -> v*Sin(v*10));
c.PlotRect := Rect(0,-3,Pi,6);
c.Graph[0].Color := Colors.Green;
c.Graph[0].Thickness := 3;
c.AddLineGraph(0,Pi,v -> Sqrt(v));
var c2 := new LineGraphWPF(0,Pi,v -> Sin(v*10)-Cos(v*7));

var xx := |1.0,2,3,4,5|;
var yy := |5.0,15,7,12,2|;
var c4 := new LineGraphWPF(xx,yy);
c4.AddMarkerGraph(xx,yy);

var gg := new GridWPF(2,2,3);
new LineGraphWPF(0,2,x->Cos(10*x));
new LineGraphWPF(0,2,x->Sqrt(x));
new LineGraphWPF(0,2,x->Sin(10*x));
new LineGraphWPF(0,2,x->exp(x));
```



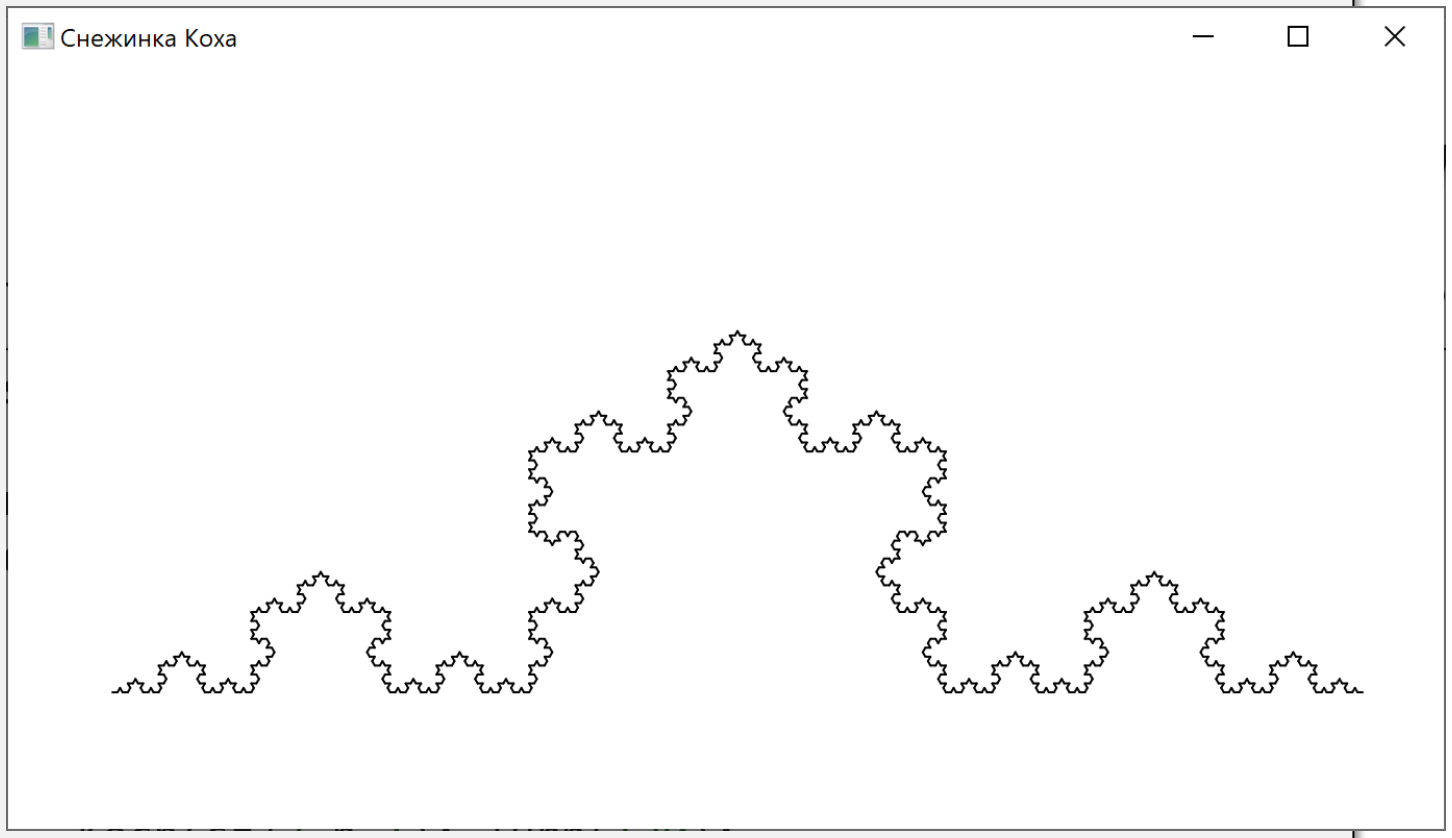
# Стандартный модуль Turtle

Стандартный модуль PlotWPF позволяет строить графики функций с различными параметрами

```
uses Turtle,GraphWPF;

procedure Koch(sz: real; n: integer);
begin
  if n = 0 then
    Forw(sz)
  else begin
    Koch(sz/3,n-1); Turn(-60);
    Koch(sz/3,n-1); Turn(120);
    Koch(sz/3,n-1); Turn(-60);
    Koch(sz/3,n-1);
  end;
end;

begin
  Window.Title := 'Снежинка Коха';
  Forw(-350);
  Down;
  Koch(600,5);
end.
```



# Это всего лишь небольшой обзор. PascalABC.NET – многогранный язык с огромным количеством возможностей

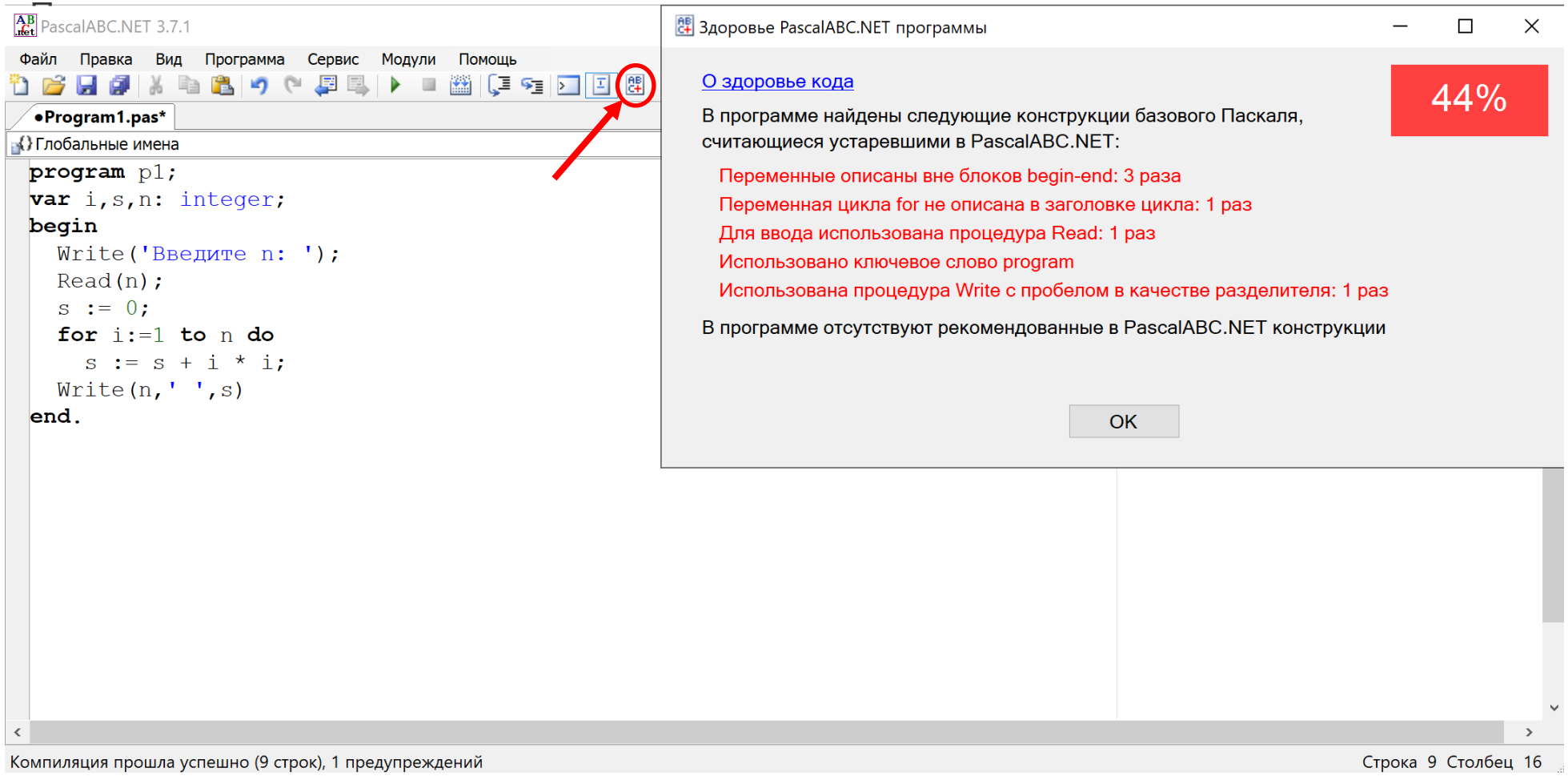
Михалкович С.С.

*Южный федеральный университет,*
*Институт математики, механики и компьютерных наук*

miks@sfedu.ru

# Не пишите плохие программы на PascalABC.NET!

**Здоровье кода PascalABC.NET** - показатель качества кода программы. Если в программе используются устаревшие конструкции, здоровье кода снижается



# Пишите хорошие программы на PascalABC.NET!

**Здоровье кода PascalABC.NET** - показатель качества кода программы. Если в программе используются новые средства PascalABC.NET, здоровье кода увеличивается