Короткий код - панацея или вред?

Демяненко Я.М.

Южный федеральный университет,
Институт математики, механики и компьютерных наук

demyana@sfedu.ru

Доклад на пятой Всероссийской научно-методической конференции
«Использование системы программирования PascalABC.NET
в обучении программированию»
(Ростов-на-Дону, 28-29 марта 2025 г.)

Чем навеяна тема?

Цепочка рассуждений

- Олимпиады
- ЕГЭ

- Сокращаем время
- Увеличиваем скорость

- Достижение цели
- Успех

Можно существенно сокращать код, не жертвуя функциональностью?

С одной стороны — чем меньше, тем лучше

В мире спортивного программирования существует разновидность необычных контестов — **соревнование на самый короткий код**

В программировании есть много концепций, которые **не имеют практического применения**:

- квайны (quine) — код, который выводит сам себя
- обфускация — практика создания кода, который трудно понять
- программы-самоликвидаторы, программы-головоломки, однострочники и т.п.
- **сайзкодинг — это когда размер программ максимально минимизируют просто потому, что могут (**размер программ от 256 байт и меньше для разных типов процессоров. Для сравнения: примерно столько занимает этот абзац, если его сохранить в однобайтной кодировке**)**

Аргументы за короткий код

Общие

- На один экран помещается больше
- Прокручивать код надо реже
- Меньше приходится набирать, соответственно код пишется быстрее
- Более короткая легче просматривается в системах контроля версий
- Вы сможете произвести впечатление на друзей и коллег

Специфика Python

- Научившись писать однострочники, вы попутно куда лучше разберетесь в основах языка Python
- Вы научитесь писать код более «питонично»

С другой стороны

Код **читается** намного **чаще**, чем **пишется**

А из этого следуют неизбежные вопросы

Неизбежные вопросы

- Насколько сложнее читать (понимать) короткий код?
- Насколько сложнее писать короткий код?
- Насколько сложнее сопровождать(редактировать) короткий код?
- Какие навыки нужны для использования короткого кода?

Признаки «хорошего» программного кода

1. **Он легко читаем**
2. **Он хорошо документирован**
3. **Он прост**
   Даже маленькие кусочки кода могут делать очень сложные вещь, однако отличный код, по мнению программистов, обычно прост
   Сложность кода прямо пропорциональна количеству багов
4. **Он гибок**
5. **Его легко поддерживать**
   Не важно, насколько хорошо написан код, неизбежно в нем могут оказаться ошибки
   Кому-то придется исправлять их. Сделайте так, чтобы это было удобно
6. **Он работает**

Попробуем разобраться: полезен или вреден короткий код?

За счёт чего можно сократить код?

PascalABC.Net

- Короткие программы – ##
- Супер короткие программы – ###
- Короткие функции Lst, LLst, HSet, SSet, Dict, KV
- Лямбды
- Срезы
- Точечная запись при вызове методов
- Запросы

Python

- Однострочники для ускорения написания кода

C++, Java, C#, Кумир и другие языки

Короткие программы –

Без констант, типов и глобальных данных!
Но! Модули, процедуры и функции разрешены

```
begin
  Writeln( 'Привет!' );
  ...
end.
```

```
##
Writeln( 'Привет!' );
...
```

Суперкороткие программы –

```
var x: integer;
begin
  x := ReadInteger;
  Print( 'Привет!' );
  ...
end.
```

```
###
var x := RI;
Pr( 'Привет!' );
...
```

```
###
var x := RI; // ReadInteger -> RI;
Pr( 'Привет!' ); // Print -> Pr
...
```

integer	int
boolean	bool
BigInteger	bi
ReadInteger	RI
ReadReal2	RR2
ReadString3	RS3
Print	Pr
Println	Prln
…	

Короткие функции Lst, LLst, HSet, SSet, Dict, KV


```
##
HSet(ReadLines('15.pas')).count.print
```

```
14
```

```
//Обозначим через ДЕЛ(n, m) утверждение «натуральное число n делится без остатка на натуральное число m». Для какого наибольшего натурального числа A формула

//¬ДЕЛ(x, A) → (ДЕЛ(x, 6) → ¬ДЕЛ(x, 9))
//тождественно истинна (то есть принимает значение 1 при любом натуральном значении переменной x)?

begin
  // Возьмём большой диапазон a: от 1 до 10000
  for var a := 10000 downto 1 do
    // Если для всех натуральных x (возьмём некоторый большой диапазон)
    // выполняется условие задачи, то мы нашли a
    if (1..100000).All(x -> not x.Divs(a) <= (x.Divs(6) <= not x.Divs(9))) then
    begin
      Print(a);
      break;
    end;
end.
```

Генераторы массивов с лямбда-функциями

Генерация по формуле a[i] = f(i)


```
begin
  var a:=new integer[10];
  for var i:=0 to 9 do
    a[i]:=(i+1)*(i+1);
  a.println
end.
```

```
##
var a := ArrGen(10,i->i*i,1);
a.print
```

Задача 2. Построение фрагмента таблицы истинности

```
##
Println( 'x y z w' );
for var x:=False to True do
 for var y:=False to True do
  for var z:=False to True do
   for var w:=False to True do
    if (x or y) and (y <> z) and not w
    then
     Println( Ord(x), Ord(y), Ord(z), Ord(w) );
```

```
 x  y  z  w  F
---------
 0  1  0  0  1
 1  0  1  0  1
 1  1  0  0  1
```


```
## uses school;
var tbl:=TrueTable((x,y,z,w)->(x or y) and (y<>z) and not w);
TrueTablePrint( tbl, 1 , 'xyzw');
```

Задача 15

Для какого наибольшего натурального числа A формула
¬ДЕЛ(x, A) → (ДЕЛ(x, 6) → ¬ДЕЛ(x, 9))
тождественно истинна (то есть принимает значение 1
при любом натуральном значении переменной x)?

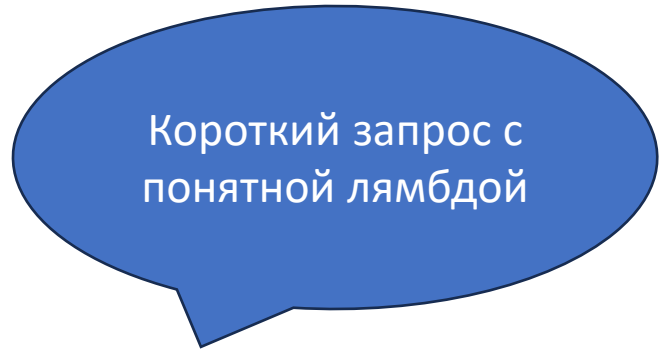
Понятно читается

```
##

  for var a := 10000 downto 1 do
    if (1..100000).All(x -> not x.Divs(a) <= (x.Divs(6) <= not x.Divs(9))) then
    begin
      Print(a);
      break;
    end;
```

Фильтрация элементов по условию (в новый массив)

```
begin
  var a := arrGen(10, i -> 2 * i + 1);
  var b := new integer[10];
  var k := 0;
  for var i := 0 to 9 do
    if a[i] mod 3 = 0 then
    begin
      b[k] := a[i];
      k += 1;
    end;
  for var i := 0 to k-1 do
    print(b[i]);
  println;
  println(k);
end.
```

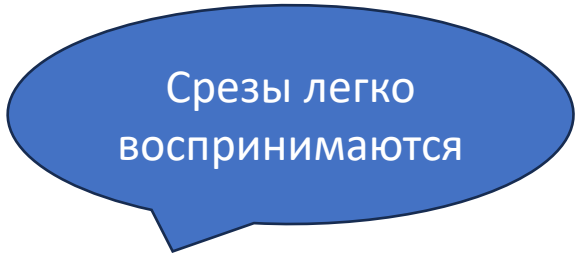



```
##
var a := arrGen(10,i->2*i+1);
var b := a.Where(x -> x mod 3 = 0);
b.println;
println(b.Count);
```

```
3 9 15
3
```

Фильтрация элементов по индексу

```
begin
  var a := arrGen(12, i -> 2 * i + 1);
  var b := new integer[10];
  var k := 0;
  for var i := 0 to 9 step 3 do
      b[k] := a[i];
      k += 1;
    end;
  for var i := 0 to k-1 do
    print(b[i]);
  println;
  println(k);
end.
```




```
##
var a := arrGen(12, i -> 2 * i + 1);
var b := a[::3];
b.println;
println(b.Count);
```

```
1 7 13 19
4
```

Задача 17

Рассматривается множество целых чисел, принадлежащих числовому отрезку [1016; 7937], которые делятся на 3 и не делятся на 7, 17, 19, 27.

Найдите количество таких чисел и максимальное из них.
В ответе запишите два целых числа: сначала количество, затем максимальное число.

Задача 17. Решение 1

Формальный подход, соединение алгоритмов, длинная запись условия

```
begin
  var count := 0;
  var max := -MaxInt;
  for var x := 1016 to 7937 do
    if (x mod 3 = 0) and (x mod 7 <> 0) and (x mod 17 <> 0) and  (x mod 19 <> 0) and (x mod 27 <> 0) then
    begin
      count += 1;
      if x > max then
        max := x;
    end;
  Print(count,max);
end.
```

Задача 17. Решение 2

Использование методов Divs и DivsAny

```
begin
  var count := 0;
  var max := -MaxInt;
  for var x := 1016 to 7937 do
    if x.Divs(3) and not x.DivsAny(7, 17, 19, 27) then
    begin
      count += 1;
      if x > max then
        max := x;
    end;
  Print(count,max);
end.
```

Задача 17. Решение 3

Анализ условия.
Искомый максимальный элемент является последним удовлетворяющим условию

```
begin
  var count := 0;
  var last := 0;
  for var x := 1016 to 7937 do
    if x.Divs(3) and not x.DivsAny(7, 17, 19, 27) then
    begin
      count += 1;
      last := x;
    end;
  Print(count,last);
end.
```

Задача 17. Решение 4

Использование последовательностей

```
begin

  // Рассмотрим последовательность целых от 1016 до 7937,
  // делящихся на 3 и не делящихся ни на одно из 7, 17, 19, 27

  var seq := (1016..7937).Where(x -> x.Divs(3) and not x.DivsAny(7, 17, 19, 27));

  // Выведем количество элементов этой последовательности и ее максимальный элемент
  Print(seq.Count,seq.Max);

end.
```

Однострочники на Python: if

```
if 3 < 2:
  var=21
else:
  var=42
```

```
var = 21 if 3<2 else 42
```

Букв одинаковое количество, только по горизонтали

```
x = 42
if x > 42:
  print("no")
elif x == 42:
  print("yes")
else:
  print("maybe")
```

```
print("no") if x > 42 else print("yes") if x == 42 else print("maybe")
```

Где выгода?

Однострочники на Python: функции, циклы

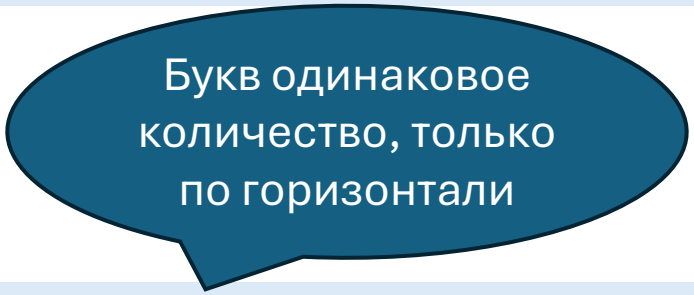
```
def f(x):
  return "hello "+ x
```

```
f = lambda x: "hello "+ x
```

```
f = exec("def f(x):\n return 'hello '+ x")
```

```
squares = []
for i in range(10):
  squares.append(i**2)
```

```
squares=[i**2 for i in range(10)]
```




```
squares = []
for i in range(10):
  if i%2==0:
    squares.append(i**2)
```

```
squares = [i**2 for i in range(10) if i%2==0]
```

Однострочники на Python: циклы

```
squares = []
for i in range(10):
  if i%2==0:
    squares.append(i**2)
  else:
    squares.append(False)
```

Букв одинаковое количество, только по горизонтали

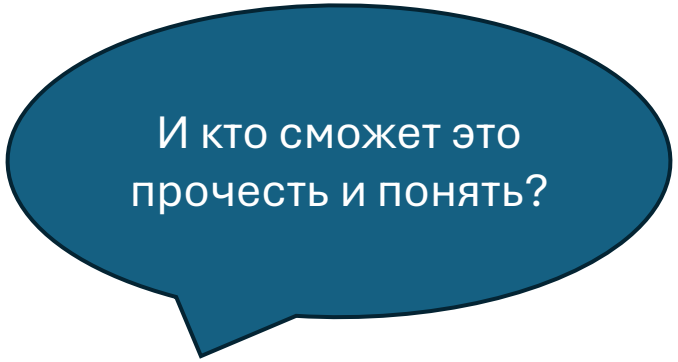
```
squares = [i**2 if i%2==0 else False for i in range(10)]
```

```
c=0
while c < 10:
  if c!=5:
    print(c)
  else:
    print("FIVE")
  c+=1
```

Но уже не так понятно

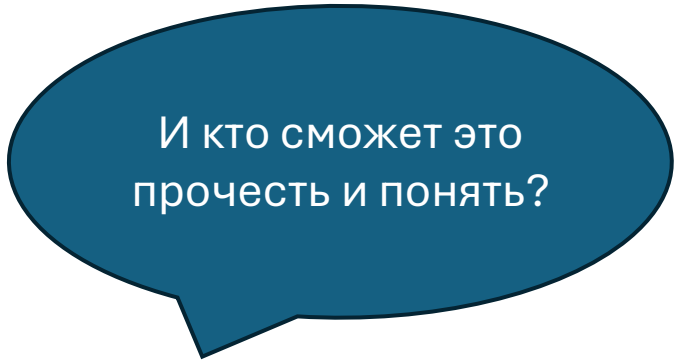
```
c =0
while c < 10: print(c)  if c!=5 else print("FIVE"); c +=1
```

Какая задача решается?




```
array = [29,99,27,41,66,28,44,78,87,19,31,76,58,88,83,97,12,21,44]
q = lambda l: q([x for x in l[1:] if x <= l[0]]) + [l[0]] + q([x for x in l if x > l[0]]) if l else []
print(q(array))
```

Оказывается, это — быстрая сортировка




```
array = [29,99,27,41,66,28,44,78,87,19,31,76,58,88,83,97,12,21,44]
q = lambda l: q([x for x in l[1:] if x <= l[0]]) + [l[0]] + q([x for x in l if x > l[0]]) if l else []
print(q(array))
```

Решение задачи ЕГЭ

Читаем данные из файла:

```
with open( "input.txt" ) as F:
  S, N = map( int, F.readline().split() )
  data = [ int(s) for s in F ]
data.sort()
```

Системный администратор создаёт архив файлов на диске, объём которого меньше, чем суммарный объём архивируемых файлов. В первой строке файла input.txt записаны два натуральных числа: S – свободное место на диске и N – количество файлов. В каждой из следующих N строк записано одно натуральное число – размер файла.
Определите максимальное число файлов, которые можно сохранить в архиве, а также максимальный размер файла, который может быть сохранён в архиве, при условии, что количество сохранённых файлов максимально.

```
print( count := max( k for k in range(1,N+1) if sum(data[:k]) <= S ) )
```

```
print( max( x for x in data[count-1:] if x <= S - sum(data[:count-1])) )
```

Краткая запись ухудшает читаемость программы и повышает вероятность ошибок!!!!!!!!

C++ Сможете прочесть?

```
for(int* p = a; p != a+10; *p++ = 0);
```

```
int a[] = {3, 7, 9, 5, 7, 2, 7};
for (int &x : a)
   x += 1;
```

```
while(pa != a + 10)
   *pc++ = *pa++ + *pb++;
```

```
for_each(begin(l), end(l), [](int i){cout << i << " ";});
```

```
std::copy_if(to_vector.begin(),to_vector.end(),std::ostream_iterator<int>(std::cout," "),[](int x) {return (x % 2) !=0;});
```

Какие задачи решают эти фрагменты кода?

```
##
var a := arrGen(10,i->2*i+1);
a.Println;
a.Count(x->x mod 3 = 0).Println;
```

```
1 3 5 7 9 11 13 15 17 19
3
```


```
##
uses school;
(2..20000).Count( x->x.IsPrime ).Println
```

```
2262
```

Решение задач через лямбды

Вывести первые 10 нечетных чисел. Посчитать, сколько среди них, кратных трём

```
var a := arrGen(10,i->2*i+1);
a.Println;
a.Count(x->x mod 3 = 0).Println;
```

```
1 3 5 7 9 11 13 15 17 19
3
```

Сколько простых в [2, 20000]?

```
##
uses school;
(2..20000).Count( x->x.IsPrime ).Println
```

```
2262
```

Проверка делимости

```
if a.D(5) or a.ND(3) then
  …
```

```
if a.Divs(5) or a.NotDivs(3) then
  …
```

```
if (a mod 5 = 0) or (a mod 3 <> 0) then
  …
```

+ эффективность кодирования
- понимание того, как все работает
- читаемость

Как устроено? Чем заменить?

```
##
function Divs(Self: integer; d: integer): boolean; extensionmethod;
begin
  Result := Self mod d = 0
end;

if 25.Divs(5) then
  Print(1);
if not 25.Divs(7) then
  Print(0)
```

```
if a mod 5 = 0 then
  Print(1);
if a mod 7 <> 0 then
  Print(0)
```

Какая задача решается?


```
## uses school;

var A := (25552..58885).Select(x -> x.Divisors)
  .Where(divs -> divs.Count(d -> d in 10..99) >= 15)
  .Select(divs -> divs.Last);

Println(A.Min, A.Max, A.Count);
```

```
25650 58800 420
```

Удалось восстановить условие задачи?

(**Е. Джобс**) Рассматривается множество целых чисел, принадлежащих числовому отрезку [25552; 58885], которые имеют не менее 15 двузначных делителей. Найдите наименьшее и наибольшее из таких чисел, а также их количество.

```
## uses school;

var A := (25552..58885).Select(x -> x.Divisors)
  .Where(divs -> divs.Count(d -> d in 10..99) >= 15)
  .Select(divs -> divs.Last);

Println(A.Min, A.Max, A.Count);
```

```
25650 58800 420
```

Попробуйте прочесть и выяснить условие

```
//Решение в функциональном стиле
begin
  ReadLines('words.txt').SelectMany(s -> s.ToWords).GroupBy(v -> v).ToDictionary(x -> x.Key, x -> x.Count).PrintLines;
end.
```

А теперь попробуйте прочесть и выяснить условие

```
begin
  var D := new Dictionary<string, integer>;
  foreach var s in ReadLines('words.txt') do
    foreach var word in s.ToWords do
      D[word] := D.Get(word) + 1;
  D.PrintLines;
end.
```

Попробуйте прочесть, зная условие задачи

```
begin
  var D := new Dictionary<string, integer>;
  foreach var s in ReadLines('words.txt') do
    foreach var word in s.ToWords do
      D[word] := D.Get(word) + 1;
  D.PrintLines;
end.
```

Построить частотный словарь слов в файле

```
//Решение в функциональном стиле
begin
  ReadLines('words.txt').SelectMany(s -> s.ToWords).GroupBy(v -> v).ToDictionary(x -> x.Key, x -> x.Count).PrintLines;
end.
```

Кратко или понятно? Вот в чём вопрос

Советы

- **Нужно различать современный стиль программирования (запросы, лямбды, срезы) и** создание короткого кода ради короткого кода
- **Не рекомендуется делать вложенные запросы**
- **Не рекомендуется в школе, да и в вузах, делать акцент на написании экстремально короткого кода**
- **Даже профессионалам не рекомендуется использовать однострочники на собеседованиях и в продакшен-коде**