Программирование на языке Паскаль

Символьные строки

Чем плох массив символов?

Это массив символов:

```
var B: array[1..N] of char;
```

- каждый символ – отдельный объект;
- массив имеет длину N, которая задана при объявлении

Что нужно:

- обрабатывать последовательность символов как единое целое
- строка должна иметь переменную длину

Описание символьных строк

В разделе *var строки* описываются следующим образом:

```
var <имя_строки>: string[<длина>];
```

***Максимальная длина строки* - 255 *символов*.**

Нумеруются ее компоненты начиная с 0, но этот нулевой байт хранит *длину строки*.

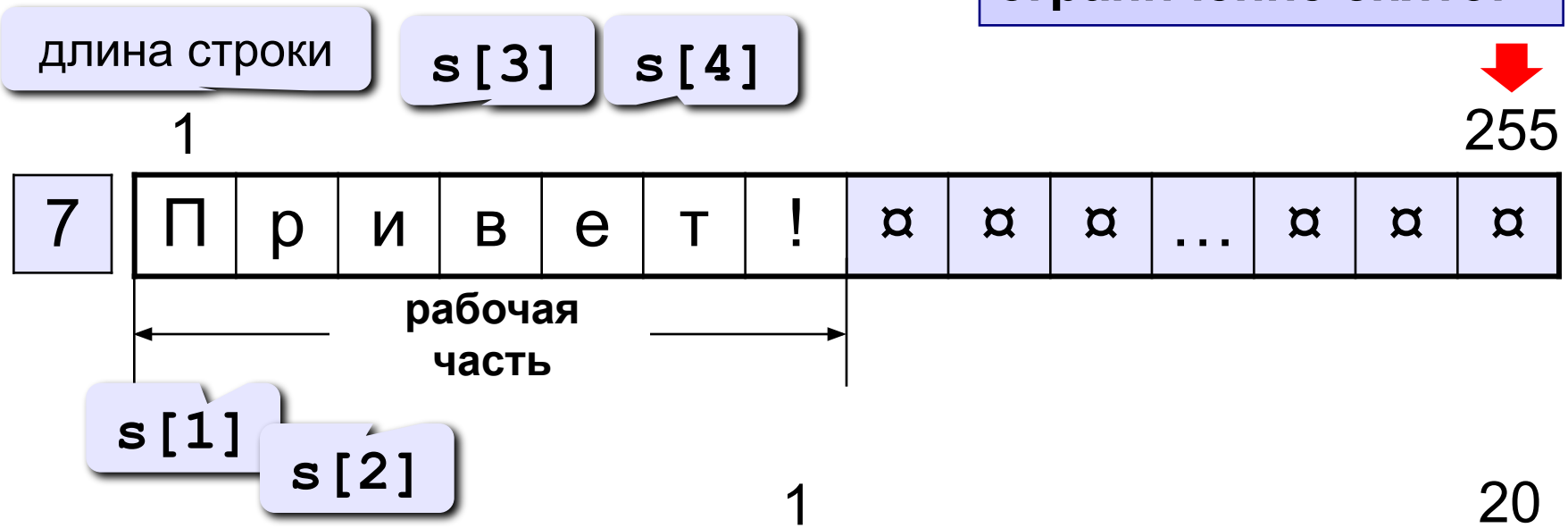
Если *<длина>* не указана, то считается, что в *строке* 255 *символов*.

Поэтому для экономии памяти следует по возможности точно указывать длину используемых строк.

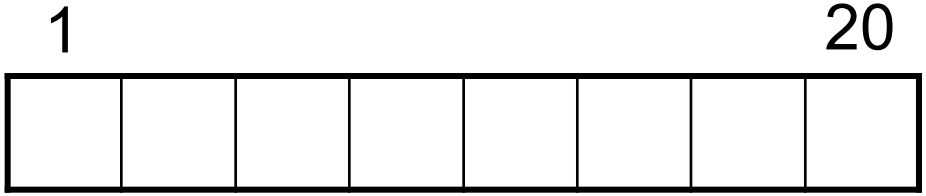
Символьные строки

```
var s: string;
```

В *Delphi* это ограничение снято!




```
var s: string[20];
```



Длина строки:

```
var n: integer;
```

```
n := length ( s );
```

Символ-константа и строка-константа: неименованные константы

В тексте программы на языке Паскаль последовательность любых символов, заключенная в *апострофы*, воспринимается как *символ* или *строка*.

Например:

```
c:='z';    {c: char}
s:='abc';    {s: string}
```

Символ-константа и строка-константа: неименованные константы

Константе автоматически присваивается "минимальный" тип данных, достаточный для ее представления:

char или *string[k]*.

Поэтому попытка написать

```
c:='zzz';          {c: char}
```

вызовет ошибку уже на этапе компиляции.

Кроме того, если константа длиннее той переменной-строки, куда ваша программа пытается ее записать, то в момент присваивания произойдет усечение ее до нужной длины.

Символ-константа и строка-константа: неименованные константы

Пустая строка **задается двумя последовательными апострофами:**

```
st:='';    { пустая строка }
```

Если же необходимо сделать так, чтобы среди символов строки содержался и сам *апостроф*, его нужно удвоить:

```
s:='Don''t worry about the apostrophe!';
```

Если теперь вывести на экран эту строку, то получится следующее:

Don't worry about the apostrophe!

Символьные строки

Задача: ввести строку с клавиатуры и заменить все буквы «а» на буквы «б».

```
program qq;
var s: string;
    i: integer;
begin
   writeln('Введите строку');
   readln(s);
   for i:=1 to Length(s) do
      if s[i] = 'а' then s[i] := 'б';
   writeln(s);
end.
```

ввод строки

длина строки

вывод строки

Задания

«3»: Ввести символьную строку и заменить все буквы «а» на буквы «б», как заглавные, так и строчные.

Пример:

```
Введите строку:
ааббссААББСС
Результат:
ббббссББББСС
```

«4»: Ввести символьную строку и заменить все буквы «а» на буквы «б» и наоборот, как заглавные, так и строчные.

Пример:

```
Введите строку:
ааббссААББСС
Результат:
ббаассББААСС
```

Задания

«5»: Ввести символьную строку и проверить, является ли она палиндромом (палиндром читается одинаково в обоих направлениях).

Пример:

```
Введите строку:
АБВГДЕ
Результат:
Не палиндром.
```

Пример:

```
Введите строку:
КАЗАК
Результат:
Палиндром.
```

Операция конкатенации

Результатом выполнения операции *конкатенации* "+", является строка, в которой исходные строки-операнды соединены в порядке их следования в выражении.

Например,

```
X:='Пример';  Y:='сложения';  Z:='строк';
Writeln(X + Y + Z);
Writeln(Y + '  ' + Z + '  ' + X);
```

На экран будут выведены строки:

Примерсложениястрок
сложения строк Пример

'Кро'+' 'код'+ 'ил'

позволит получить новую строку

'Крокодил'

Операции со строками

```
var s, s1, s2: string;
```

Запись нового значения:

```
s := 'Вася';
```

Объединение: добавить одну строку в конец другой.

```
s1 := 'Привет';
s2 := 'Вася';
s := s1 + ', ' + s2 + '!';
```

`'Привет, Вася!'`

Подстрока: выделить часть строки в другую строку.

```
s := '123456789';

s1 := Copy ( s, 3, 6 );
s2 := Copy ( s1, 2, 3 );
```

с 3-его символа

6 штук

`'345678'`

`'456'`

Удаление и вставка

Удаление части строки:

```
s := '123456789';
Delete ( s, 3, 6 );
```

6 штук

'123456789'

'129'

строка меняется!

с 3-его символа

Вставка в строку:

```
s := '123456789';
Insert ( 'ABC', s, 3 );

Insert ( 'Q', s, 5 );
```

начиная с 3-его символа

'12ABC3456789'

что вставляем

куда вставляем

'12ABQC3456789'

Поиск в строке

Поиск в строке:

s[3]

var n: integer;

```
s := 'Здесь был Вася.';
n := Pos ( 'e', s );
if n > 0 then
     writeln('Буква e – это s[', n, ']')
else writeln('Не нашли');
n := Pos ( 'Вася', s );
s1 := Copy ( s, n, 4 );
```

3

n = 11

Особенности:

- функция возвращает номер символа, с которого начинается образец в строке
- если слова нет, возвращается 0
- поиск с начала (находится **первое** слово)

Примеры

```
s := 'Вася Петя Митя';
n := Pos ( 'Петя', s );
Delete ( s, n, 4 );
Insert ( 'Лена', s, n );
```

```
6
'Вася  Митя'
'Вася Лена Митя'
```

```
s := 'Вася Петя Митя';
n := length ( s );
s1 := Copy ( s, 1, 4 );
s2 := Copy ( s, 11, 4 );
s3 := Copy ( s, 6, 4 );
s := s3 + s1 + s2;
n := length ( s );
```

```
14
'Вася'
'Митя'
'Петя'
'ПетяВасяМитя'
12
```

Пример решения задачи

Задача: Ввести имя, отчество и фамилию. Преобразовать их к формату «фамилия-инициалы».

Пример:

```
Введите имя, фамилию и отчество:
Василий Алибабаевич Хрюндиков
Результат:
Хрюндиков В.А.
```

Алгоритм:

- найти первый пробел и выделить имя
- удалить имя с пробелом из основной строки
- найти первый пробел и выделить отчество
- удалить отчество с пробелом из основной строки
- «сцепить» фамилию, первые буквы имени и фамилии, точки, пробелы…

Программа

```
program qq;
var s, name, otch: string;
    n: integer;
begin
   writeln('Введите имя, отчество и фамилию');
   readln(s);
   n := Pos(' ', s);
   name := Copy(s, 1, n-1); { вырезать имя }
   Delete(s, 1, n);
   n := Pos(' ', s);
   otch := Copy(s, 1, n-1); { вырезать отчество }
   Delete(s, 1, n);         { осталась фамилия }
   s := s + ' ' + name[1] + '.' + otch[1] + '.';
   writeln(s);
end.
```

Задания

«3»: Ввести в одну строку фамилию, имя и отчество, разделив их пробелом. Вывести инициалы и фамилию.

Пример:

```
Введите фамилию, имя и отчество:
Иванов Петр Семёнович
Результат:
П.С. Иванов
```

«4»: Ввести имя файла (возможно, без расширения) и изменить его расширение на «`.exe`».

Пример:

```
Введите имя файла:
qqq
Результат:
qqq.exe
```

```
Введите имя файла:
qqq.com
Результат:
qqq.exe
```

Задания

«5»: Ввести путь к файлу и «разобрать» его, выводя каждую вложенную папку с новой строки

Пример:

```
Введите путь к файлу:
C:\Мои документы\10-Б\Вася\qq.exe
Результат:
C:
Мои документы
10-Б
Вася
qq.exe
```

Задачи на обработку строк

Задача: с клавиатуры вводится символьная строка, представляющая собой сумму двух целых чисел, например:

```
12+35
```

Вычислить эту сумму:

```
12+35=47
```

Алгоритм:

1) найти знак «+»
2) выделить числа слева и справа в отдельные строки
3) перевести строки в числа
4) сложить
5) вывести результат

Преобразования «строка»-«число»

Из строки в число:

```
s := '123';
Val ( s, N, r ); { N = 123 }
  { r = 0, если ошибки не было
    r – номер ошибочного символа}
s := '123.456';
Val ( s, X, r ); { X = 123.456 }
```

```
var N, r: integer;
    X: real;
    s: string;
```

Из числа в строку:

```
N := 123;
Str ( N, s );        { '123' }
X := 123.456;
Str ( X, s );        { '1.234560E+002' }
Str ( X:10:3, s ); { '   123.456' }
```

Программа

```
program qq;
var s, s1, s2: string;
    r, n, n1, n2, sum: integer;
begin
  writeln('Введите выражение (сумму чисел):');
  readln(s);
  n:=Pos('+', s);
  s1:=Copy(s, 1, n-1);
  s2:=Copy(s, n+1, Length(s)-n);
  Val(s1, n1, r);
  Val(s2, n2, r);
  sum:=n1+n2;
  writeln(n1, '+', n2, '=', sum);
end.
```

слагаемые-строки

сумма

слагаемые-числа

слагаемые-строки

слагаемые-числа

Задания

«3»: **Ввести арифметическое выражение: разность двух чисел. Вычислить эту разность.**

Пример:

```
25-12
Ответ: 13
```

«4»: **Ввести арифметическое выражение: сумму трёх чисел. Вычислить эту сумму.**

Пример:

```
25+12+34
Ответ: 71
```

Задания

«5»: Ввести арифметическое выражение с тремя числами, в котором можно использовать сложение и вычитание. Вычислить это выражение.

Пример:

```
25+12+34
```

```
Ответ: 71
```

Пример:

```
25+12-34
```

```
Ответ: 3
```

Пример:

```
25-12+34
```

```
Ответ: 47
```

Пример:

```
25-12-34
```

```
Ответ: -21
```

Задания

«6»: Ввести арифметическое выражение с тремя числами, в котором можно использовать сложение, вычитание и умножение. Вычислить это выражение.

Пример:

```
25+12*3
```

Ответ: 61

Пример:

```
25*2-34
```

Ответ: 16

Пример:

```
25-12+34
```

Ответ: 47

Пример:

```
25*2*3
```

Ответ: 150

Посимвольный ввод

Задача: с клавиатуры вводится число N, обозначающее количество футболистов команды «Шайба», а затем – N строк, в каждой из которых – информация об одном футболисте таком формате:

<Фамилия> <Имя> <год рождения> <голы>

Все данные разделяются одним пробелом. Нужно подсчитать, сколько футболистов, родившихся в период с 1988 по1990 год, не забили мячей вообще.

Алгоритм:

```
for i:=1 to N do begin
  { пропускаем фамилию и имя }
  { читаем год рождения Year и число голов Gol }
  if (1988 <= Year) and (Year <=1990) and
     (Gol = 0) then  { увеличиваем счетчик }
end;
```

Посимвольный ввод

Пропуск фамилии:

```
var c: char;
```

```
repeat
  read(c);
until c = ' '; { пока не встретим пробел }
```

Пропуск имени:

```
repeat read(c); until c = ' ';
```

Ввод года рождения:

```
var Year: integer;
```

```
read(Year); { из той же введенной строки }
```

Ввод числа голов и переход к следующей строке:

```
readln(Gol); { читать все до конца строки }
```

```
var Gol: integer;
```

Программа

```
program qq;
var c: char;
    i, N, count, Year, Gol: integer;
begin
  writeln('Количество футболистов');
  readln(N);
  count := 0;
  for i:=1 to N do begin
    repeat read(c); until c = ' ';
    repeat read(c); until c = ' ';
    read(Year);
    readln(Gol);
    if (1988 <= Year) and (year <= 1990) and
       (Gol = 0) then count := count + 1;
  end;
  writeln(count);
end.
```

Посимвольный ввод

Если фамилия нужна:

```
var fam: string;
```

```
fam := '';   { пустая строка }
repeat
  read(c);              { прочитать символ }
  fam := fam + c;       { прицепить к фамилии }
until c = ' ';
```

Вместо `read(Year)`:

```
var s: string;
```

```
s := '';   { пустая строка }
repeat
  read(c);              { прочитать символ }
  s := s + c;           { прицепить к году }
until c = ' ';
Val(s, Year, r); { строку – в число }
```

Посимвольный ввод

Если нужно хранить все фамилии:

массив символьных строк

```
const MAX = 100;
var fam: array[1..MAX] of string;
...
fam[i] := '';  { пустая строка }
repeat
  read(c);          { прочитать символ }
  fam[i] := fam[i] + c;
until c = ' ';
```

Задания

Информация о футболистах вводится так же, как и для приведенной задачи (сначала N, потом N строк с данными).

«3»: Вывести фамилии и имена всех футболистов, которые забили больше двух голов.

Пример:

```
Иванов Василий
Семёнов Кузьма
```

«4»: Вывести фамилию и имя футболиста, забившего наибольшее число голов, и количество забитых им голов.

Пример:

```
Иванов Василий 25
```

Задания

«5»: Вывести *в алфавитном порядке* фамилии и имена всех футболистов, которые забили хотя бы один гол. В списке не более 100 футболистов.

Пример:

```
Васильев Иван
Иванов Василий
Кутузов Михаил
Пупкин Василий
```