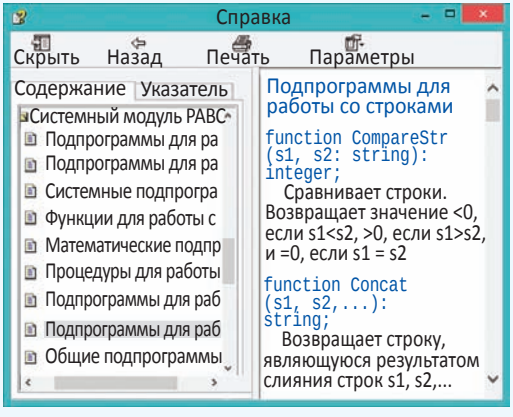
§ 8. Стандартные процедуры и функции для работы со строковыми величинами

Пример 8.1. В справке среды программирования PascalABC.NET в разделе **Системный модуль PascalABC-System → Подпрограммы для работы со строками** можно найти описание функций и процедур:

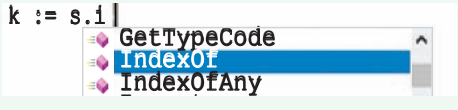


Пример 8.2. Примеры использования функций.

`d := Length ('Компьютер');`	d = 9
`s := 'Строка';` `d := Length(s);`	d = 6

`s := 'Не слово хозяин хозяину, а хозяин слову хозяин';`
`s1 := 'хозяин';`
`d := Length(s);`
`N1 := Pos(s1,s);`
`N2 := LastPos(s1,s);`
`N3 := PosEx(s1,s,15);`
	d = 46
	N1 = 10
	N2 = 41
	N3 = 17

PascalABC позволяет обращаться к функциям обработки строк и по-другому: если после имени строковой переменной поставить точку, то появится список функций.



8.1. Поиск в строке

Современные компьютерные устройства позволяют достаточно быстро осуществлять поиск в тексте, используя для этого различные алгоритмы. Языки программирования предоставляют широкий набор функций для работы с текстом. Некоторые функции языка программирования Pascal для поиска подстроки (части строки) в другой строке представлены в таблице.

Функция	Описание
`Length(s)`	Определяет длину строки `s` (количество символов в строке)
`Pos(s1, s)`	Определяет позицию подстроки `s1` в строке `s`. Если не найдена — возвращает 0
`LastPos (s1, s)`	Определяет позицию последнего вхождения подстроки `s1` в строке `s`. Если не найдена — возвращает 0
`PosEx (s1,s,from)`	Определяет позицию подстроки `s1` в строке `s`, начиная с позиции `from`. Если не найдена — возвращает 0

Подробное описание функций и процедур для работы со строками можно найти в справочной системе PascalABC.NET (пример 8.1), а также в *Приложении 2* (см. с. 161—162).

В примере 8.2. показано, как применять указанные функции.

Пример 8.3. Написать программу, которая вводит слово, а затем выводит его по одному символу в строке.

Этапы выполнения задания

I. Исходные данные: переменная s — исходное слово.

II. Результат: слово на экране, каждый символ в отдельной строке.

III. Алгоритм решения задачи.

1. Вводим исходные данные.

2. Определяем длину слова. Переменной n присваиваем значение функции length(s).

3. В цикле **for** выводим по одному символы введенного слова.

IV. Описание переменных: s – string, n – integer.

Пример 8.4. Написать программу, которая выводит на экран последний символ введенного слова и определяет, встречается ли этот символ в слове еще раз. Если встречается, то программа выводит индекс символа.

Этапы выполнения задания

I. Исходные данные: переменная s — введенное слово.

II. Результат: последний символ в слове и соответствующее сообщение — встречается или не встречается.

III. Алгоритм решения задачи.

1. Вводим исходные данные.

2. Определяем длину слова. Переменной n присваиваем значение функции length(s).

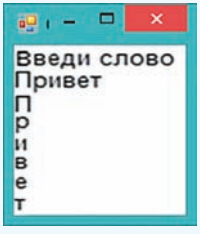
3. Определяем последний символ. Поскольку символы в строке нумеруются с 1, то номер последнего символа совпадает с длиной строки.

Пример 8.3.

V. Программа:

```
uses GraphABC;
var
  s: string; n: integer;
begin
  writeln('Введи слово');
  readln(s); writeln(s);
  n := length(s);
  for var i := 1 to n do
    writeln(s[i]);
end.
```

VI. Тестирование.



Пример 8.4.

V. Программа:

```
var s: string; n, k: integer;
begin
  writeln('Введи слово');
  readln(s); n := length(s);
  writeln('Последний символ - ',
s[n]);
  k := pos(s[n], s);
  if k = n then
    writeln('Символ один')
  else
   writeln('Символ с индексом ', k)
end.
```

VI. Тестирование. Результаты:

Окно вывода
Введи слово Строка Последний символ - а Символ один

Окно вывода
Введи слово Информатика Последний символ - а Символ на месте 7

Один из самых известных и эффективных алгоритмов поиска подстроки в строке — алгоритм Кнута — Морриса — Пратта (КМП-алгоритм). Алгоритм был разработан Д. Кнутом и В. Праттом и (независимо от них) Д. Моррисом. Результаты своей работы они опубликовали совместно в 1977 г. Время работы алгоритма линейно зависит от объема входных данных.

Дональд Кнут (род. в 1938)

Вон Пратт (род. в 1944)

Джеймс Моррис (род. в 1941)

Дональд Эрвин Кнут — американский ученый, автор известной серии книг об основных алгоритмах и методах вычислительной математики.

Вон Рональд Пратт — почетный профессор в Стэнфордском университете, известен своим вкладом в развитие таких областей информатики, как алгоритмы поиска и сортировок, а также тестирование простоты чисел.

Джеймс Хирэм Моррис — американский профессор. Занимался разработкой в области языков программирования и технологическим дизайном программных продуктов.

Пример 8.5. Примеры использования функций.

```
s := 'Информатика';
s := UpperCase(s);
```
После преобразования в строке `s` будет записано `ИНФОРМАТИКА`

```
s := 'Информатика';
s[1] := LowCase(s[1]);
```
После преобразования в строке `s` будет записано `информатика`

4. Определяем позицию последнего символа в слове. Переменной `k` присваиваем значение функции `pos` для последнего символа. Если оно равно длине строки, то символ в слове единственный, иначе в слове есть другой такой же символ.

5. Выводим результат.

IV. Описание переменных: `s` — `string`, `n`, `k` — `integer`.

Поиск текстовой информации во всемирной паутине заключается в том, чтобы по запросу пользователя найти документы, содержащие указанные ключевые слова. Для этого различные поисковые системы используют разные алгоритмы. Запрос, который вводит пользователь, может содержать строчные и заглавные буквы. Для осуществления поиска буквы в слове обычно приводят к одному регистру: либо все строчные, либо все заглавные. В Pascal также есть функции преобразования.

Функция	Описание
`LowCase(c)`	Преобразует один символ (букву) в строчную букву
`LowerCase(s)`	Преобразует все символы (буквы) строки в строчные буквы
`UpCase(c)`	Преобразует один символ (букву) в заглавную букву
`UpperCase(s)`	Преобразует все символы (буквы) строки в заглавные буквы

В примере 8.5 показано применение этих функций.

8.2. Копирование, вставка и удаление символов

При работе с текстом в текстовом редакторе часто приходится пользоваться буфером обмена. Часть текста (подстрока) копируется (вырезается) в буфер обмена, а затем вставляется в другое место в тексте. В языке Pascal реализованы команды для работы с фрагментом текста, которые представлены в таблице.

Команда	Описание
`Copy (s,index,count)`	Функция копирует часть строки `s` в другую строку
`Delete (s,index,count);`	Процедура удаляет символы строки `s`
`Insert (s1,s,index);`	Процедура вставляет подстроку `s1` в строку `s`

Во всех командах переменная `s` обозначает исходную строку, над которой производится операция. Переменная `index` обозначает позицию символа, начиная с которого выполняют операцию, а переменная `count` — количество символов. Разберем команды подробнее (пример 8.6).

Команда `copy` является функцией, и результат ее работы присваивается другой переменной. Строка `s` при этом не изменяется.

```
s1 := copy(s, index, count);
```

Команды `delete` и `insert` являются процедурами, они изменяют строку `s`. В примере 8.7 показано, как применяются указанные команды.

Пример 8.6. Команды для преобразования строк.

1. Запись `s1 := Copy(s, index, count);` означает, что в строке `s` выделяют `count` символов, выделение начинают с символа, индекс которого записан в переменной `index`. Эти символы копируются в переменную `s1` (действие команды сравнимо с копированием фрагмента текста в буфер обмена).

2. Запись `Delete(s, index, count);` означает, что в строке `s` выделяют `count` символов, выделение начинают с символа, индекс которого записан в переменной `index`. Выделенные символы удаляются из строки `s`. Остальные символы строки сдвигаются влево (действие команды сравнимо с удалением фрагмента текста).

3. Запись `Insert(s1, s, index);` означает, что в строку `s` вставляют символы строки `s1`, вставка происходит в позиции `index`. Остальные символы строки сдвигаются вправо (действие команды сравнимо со вставкой фрагмента текста из буфера обмена).

Пример 8.7. Примеры использования команд.

`s := 'Информатика';`	
`s1 := copy(s,3,5);`	`s1 = 'форма'`
`Delete(s,8,4);`	`s = 'Информа'`
`Insert ('ция',s,8);`	`s = 'Информация'`

Пример 8.8.

V. Программа:

```
var  s, p, t: string;
     n1, n2, k: integer;
begin
  writeln('Строка s');
  readln(s);
  writeln('Подстрока p');
  readln(p);
  n1 := length(s);
  n2 := length(p);
  k := 0;
  for var i := 1 to n1 - n2 + 1 do
  begin
    t := copy(s, i, n2);
    if t = p then
      k := k + 1;
  end;
  writeln('Встречается ', k,
' раз(-а)');
  end.
```

VI. Тестирование.

Запустить программу, ввести строку «Не слово хозяин хозяину, а хозяин слову хозяин» и подстроку «хозяин». Результат:

Окно вывода

```
Строка s
Не слово хозяин хозяину, а хозяин слову хозяин
Подстрока p
хозяин
Встречается 4 раз(-а)
```

Если для той же строки ввести подстроку «хозяйка», то результат будет таким:

Окно вывода

```
Строка s
Не слово хозяин хозяину, а хозяин слову хозяин
Подстрока p
хозяйка
Встречается 0 раз(-а)
```

Пример 8.8. Написать программу, которая определит, сколько раз заданная подстрока встречается в строке.

Этапы выполнения задания

I. Исходные данные: переменная s — исходная строка, p — исходная подстрока.

II. Результат: k — искомое количество.

III. Алгоритм решения задачи.

1. Вводим исходные данные.

2. Инициализируем значение счетчика k := 0;

3. Определяем длины n1 и n2 для строки s и подстроки p.

4. В цикле **for** от 1 до разницы в длинах строки s и подстроки p:

4.1. Выделяем из строки s подстроку t такой же длины, что и длина p, начиная с текущего символа.

4.2. Сравниваем подстроки. Если они равны, то увеличиваем значение счетчика на 1.

5. Выводим результат.

IV. Описание переменных: s, p, t – string, n1, n2, k – integer.

Пример 8.9. Написать программу, которая из слова ТЕСТИРОВАНИЕ получит слово РИСОВАНИЕ, используя процедуры и функции преобразования строк.

Этапы выполнения задания

I. Исходные данные: слово ТЕСТИРОВАНИЕ будем хранить как константу с именем s.

II. Результат: полученное слово.

III. Алгоритм решения задачи.

1. Результат не зависит от вводимых данных.

2. В строку S1 запишем шестой символ исходной строки.

3. Скопируем из строки S восемь символов, начиная с позиции 5. Добавим к строке S1. Получим 'РИРОВАНИЕ'.

4. В полученной строке удалим третий символ ('РИОВАНИЕ').

5. Вставим на третье место третий символ исходной строки ('РИСОВАНИЕ').

6. Выведем результат.

IV. Описание переменных: S1 — string.

Пример 8.9.

V. Программа:

```
const S = 'ТЕСТИРОВАНИЕ';
var S1: string;
begin
 S1 := S[6]; S1 := S1 + copy(S,5,8);
 //РИРОВАНИЕ
 delete(S1, 3, 1);
 //РИОВАНИЕ
 insert(S[3], S1, 3);
 //РИСОВАНИЕ
 writeln('Слово =', S1);
end.
```

VI. Тестирование.

Окно вывода

```
Слово = РИСОВАНИЕ
```

Для копирования последних 8 символов из строки S можно использовать функцию RightStr: S1 := S1 + RightStr(S,8).

Для копирования первых символов из строки можно использовать функцию LeftStr.

1. Что такое длина строки?
2. С помощью какой функции можно найти длину строки?
3. Какие функции используются для определения позиции подстроки в строке?
4. Как скопировать символы из одной строки в другую?
5. Какая процедура используется для удаления символов из строки?
6. Какая процедура предназначена для вставки символов в строку?
7*. Сопоставьте команды преобразования строк с операциями, выполняемыми с помощью буфера обмена.

Упражнения

1 В программу из примера 8.3 внесли следующие изменения:

```
for var i := 1 to n do
 begin
  write(s[i]);
  if i mod 2 = 0 then
   writeln;
 end;
```

Как теперь выводится слово? Объясните почему.

2 Измените программу из примера 8.3 так, как указано ниже.

1. Каждая буква должна выводиться своим цветом (можно использовать случайное задание цветов).

2. Буквы, стоящие на четных местах, должны выводиться одним цветом, а на нечетных — другим.

3 Измените программу из примера 8.4 так, чтобы на экран выводился символ введенного слова, стоящий посередине (для слов с четным количеством букв — символ справа от середины).

1. Проверьте правильность работы своей программы на предложенных примерах. Откройте файл с таблицей и запишите результаты.

Слово	Результат
Школа	о
гимназия	а
форма	
Интернет	

2. Допишите в таблицу два своих примера.
3. Что будет выведено, если ничего не вводить, просто нажать Enter?
4. Проверьте, встречается ли выведенный символ в слове еще раз.
5. Выведите позиции всех символов, совпадающих с символом слова, находящимся посередине.

4 Измените программу из примера 8.4 так, чтобы строчные и заглавные буквы анализировались программой одинаково (например, для слова «Анна» ответ должен быть следующим: «`Последний символ – а, символ встретился на месте 1`»).

5 Даны два слова. Верно ли, что одно из слов начинается на ту же букву, на которую заканчивается другое? (Первая буква одного из слов может быть заглавной.) Если да, то вывести букву, иначе — соответствующее сообщение.

1. Проверьте правильность работы своей программы на предложенных примерах. Откройте файл с таблицей и запишите результаты.

Слово 1	Слово 2	Результат
array	yellow	y
apple	auto	неверно
Рыба	Арбуз	

2. Допишите в таблицу два своих значения.
3*. Если ответ «верно», указать, принадлежат ли буквы одному регистру.

6 Измените программу из примера 8.8 так, чтобы при `k = 0` выводилось сообщение `'Подстрока в строке не встречается'`.

7 Программу из примера 8.8 записали следующим образом:

```
var s, p, t: string;
  k, i: integer;
begin
 writeln('Строка s');
 readln(s);
 writeln('Подстрока p');
 readln(p);
 k := 0; i := 1;
 while PosEx(p, s, i) <> 0 do
 begin
  i := PosEx(p, s, i) + 1; k := k + 1;
 end;
 writeln('Встречается', k, 'раз(-а)');
end.
```

Сравните эту программу и программу из примера 8.8, определив, сколько раз выполнится команда цикла в каждой из программ для перечисленных случаев.

1. Строка s := «Не слово хозяин хозяину, а хозяин слову хозяин», подстрока p := «хозяин».
2. Строка s := «Не слово хозяин хозяину, а хозяин слову хозяин», подстрока p := «хозяйка».

*Можно ли подобрать тест, для которого количество выполнений цикла будет одинаковым для обеих программ? Если да, то какой?

8 Получите из слова «ТЕСТИРОВАНИЕ» указанные слова. Для этого используйте команды: `copy`, `delete`, `insert` и операцию сложения строк.

1. РОСТ.
2. НИВА.
3. ТОВАР.
4. ТОСТЕР.
5. ОТВЕРСТИЕ.
6. Придумайте свое слово.

§ 9. Составление алгоритмов обработки строковых величин

9.1. Анализ текста на наличие различных символов

Современные текстовые редакторы позволяют получить статистику по символам и словам в документе (пример 9.1).

Грамотный набор текста предполагает наличие только одного пробела между словами. В правильно набранном тексте количество слов будет на единицу больше, чем количество пробелов.

Пример 9.1. Статистика в документе Word (соответствующая команда на вкладке **Рецензирование**).