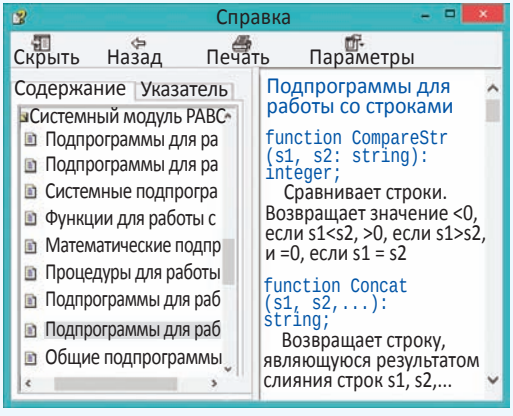
§ 8. Стандартныя працэдуры і функцыі для работы з радковымі велічынямі

Прыклад 8.1. У даведцы асяроддзя праграмірования PascalABC.NET у раздзеле **Системный модуль PascalABCSystem → Подпрограммы для работы со строками** можна знайсці апісанне функцый і працэдур:

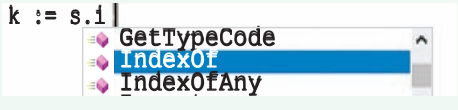


Прыклад 8.2. Прыклады выкарыстання функцый.

`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 дазваляе звяртацца да функцый апрацоўкі радкоў і па-іншаму: калі пасля імені радковай пераменнай паставіць кропку, то з'явіцца спіс функцый.



8.1. Пошук у радку

Сучасныя камп'ютарныя ўстройствы дазваляюць дастаткова хутка ажыццяўляць пошук у тэксце, выкарыстоўваючы для гэтага розныя алгарытмы. Мовы праграмірования даюць шырокі набор функцый для работы з тэкстам. Некаторыя функцыі мовы праграмірования Pascal для пошуку падрадка (часткі радка) у іншым радку паказаны ў табліцы.

Функцыя	Апісанне
`Length(s)`	Вызначае даўжыню радка `s` (колькасць сімвалаў у радку)
`Pos(s1, s)`	Вызначае пазіцыю падрадка `s1` у радку `s`. Калі не знойдзена — вяртае 0
`LastPos (s1, s)`	Вызначае пазіцыю апошняга ўваходжання падрадка `s1` у радку `s`. Калі не знойдзена — вяртае 0
`PosEx (s1,s,from)`	Вызначае пазіцыю падрадка `s1` у радку `s`, пачынаючы з пазіцыі `from`. Калі не знойдзена — вяртае 0

Падрабязнае апісанне функцый і працэдур для работы з радкамі можна знайсці ў даведачнай сістэме PascalABC.NET (прыклад 8.1), а таксама ў *Дадатку* 2 (гл. с. 161—162).

У прыкладзе 8.2 паказана, як ужываць дадзеныя функцыі.

Прыклад 8.3. Напісаць праграму, якая ўводзіць слова, а затым выводзіць яго па адным сімвале ў радку.

Этапы выканання задання

I. Зыходныя даныя: пераменная s — зыходнае слова.

II. Вынік: слова на экране, кожны сімвал у асобным радку.

III. Алгарытм рашэння задачы.

1. Уводзім зыходныя даныя.

2. Вызначаем даўжыню слова. Пераменнай n прысвойваем значэнне функцыі length(s).

3. У цыкле **for** выводзім па адным сімвалы ўведзенага слова.

IV. Апісанне пераменных: s — string, n — integer.

Прыклад 8.4. Напісаць праграму, якая выводзіць на экран апошні сімвал уведзенага слова і вызначае, ці сустракаецца гэты сімвал у слове яшчэ раз. Калі сустракаецца, то праграма выводзіць індэкс сімвала.

Этапы выканання задання

I. Зыходныя даныя: пераменная s — уведзеннае слова.

II. Вынік: апошні сімвал у слове і адпаведнае паведамленне — сустракаецца ці не сустракаецца.

III. Алгарытм рашэння задачы.

1. Уводзім зыходныя даныя.

2. Вызначаем даўжыню слова. Пераменнай n прысвойваем значэнне функцыі length(s).

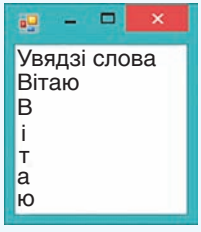
3. Вызначаем апошні сімвал. Паколькі сімвалы ў радку нумаруюцца з 1, то нумар апошняга сімвала супадае з даўжынёй радка.

Прыклад 8.3.

V. Праграма:

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

VI. Тэсціраванне.



Прыклад 8.4.

V. Праграма:

```
var s: string; n, k: integer;
begin
  writeln('Увядзі слова');
  readln(s); n := length(s);
  writeln('Апошні сімвал - ',
s[n]);
  k := pos(s[n], s);
  if k = n then
    writeln('Сімвал адзін')
  else
   writeln('Сімвал з індэксам ', k)
end.
```

VI. Тэсціраванне. Вынік.

Окно вывода

```
Увядзі слова
Радок
Апошні сімвал - к
Сімвал адзін
```

Окно вывода

```
Увядзі слова
Інфарматыка
Апошні сімвал - а
Сімвал з індэксам 4
```

Адзін з самых вядомых і эфектыўных алгарытмаў пошуку падрадка ў радку — алгарытм Кнута — Морыса — Прата (КМП-алгарытм). Алгарытм быў распрацаваны Д. Кнутам і В. Пратам і (незалежна ад іх) Д. Морысам. Вынікі сваёй працы яны апублікавалі сумесна ў 1977 г. Час работы алгарытму лінейна залежыць ад аб'ёму ўваходных даных.

Дональд Кнут (нар. у 1938)

Вон Прат (нар. у 1944)

Джэймс Морыс (нар. у 1941)

Дональд Эрвін Кнут — амерыканскі вучоны, аўтар вядомай серыі кніг пра асноўныя алгарытмы і метады вылічальнай матэматыкі.

Вон Рональд Прат — ганаровы прафесар у Стэнфардскім універсітэце, вядомы сваім укладам у развіццё такіх галін інфарматыкі, як алгарытмы пошуку і сартаванняў, а таксама тэсціраванне прастаты лікаў.

Джэймс Хірэм Морыс — амерыканскі прафесар. Займаўся распрацоўкай у галіне моў праграміравання і тэхналагічным дызайнам праграмных прадуктаў.

Прыклад 8.5. Прыклады выкарыстання функцый.

```
s := 'Інфарматыка';
s := UpperCase(s);
```

Пасля пераўтварэння ў радку `s` будзе запісана `ІНФАРМАТЫКА`

```
s := 'Інфарматыка';
s[1] := LowCase(s[1]);
```

Пасля пераўтварэння ў радку `s` будзе запісана `інфарматыка`

4. Вызначаем пазіцыю апошняга сімвала ў слове. Пераменнай `k` прысвойваем значэнне функцыі `pos` для апошняга сімвала. Калі яно роўнае даўжыні радка, то сімвал у слове адзіны, інакш у слове ёсць іншы такі ж сімвал.

5. Выводзім вынік.

IV. Апісанне пераменных: `s` – `string`, `n`, `k` – `integer`.

Пошук тэкставай інфармацыі ў сусветнай павуціне заключаецца ў тым, каб па запыце карыстальніка знайсці дакументы, якія змяшчаюць пэўныя ключавыя словы. Для гэтага розныя пошукавыя сістэмы выкарыстоўваюць розныя алгарытмы. Запыт, які ўводзіць карыстальнік, можа змяшчаць маленькія і вялікія літары. Для ажыццяўлення пошуку літары ў слове звычайна прыводзяць да аднаго рэгістра: або ўсе маленькія, або ўсе вялікія. У Pascal таксама ёсць функцыі пераўтварэння.

Функцыя	Апісанне
`LowCase(c)`	Пераўтварае адзін сімвал (літару) у малую літару
`LowerCase(s)`	Пераўтварае ўсе сімвалы (літары) радка ў малыя літары
`UpCase(c)`	Пераўтварае адзін сімвал (літару) у вялікую літару
`UpperCase(s)`	Пераўтварае ўсе сімвалы (літары) радка ў вялікія літары

У прыкладзе 8.5 паказана выкарыстанне гэтых функцый.

8.2. Капіраванне, устаўка і выдаленне сімвалаў

Пры рабоце з тэкстам у тэкставым рэдактары часта даводзіцца карыстацца буферам абмену. Частка тэксту (падрадок) капіруецца (выразаецца) у буфер абмену, а затым устаўляецца ў іншае месца ў тэксце. У мове Pascal рэалізаваны каманды для работы з фрагментам тэксту, якія пададзены ў табліцы.

Каманда	Апісанне
`Copy (s,index,count)`	Функцыя капіруе частку радка `s` у іншы радок
`Delete (s,index,count);`	Працэдура выдаляе сімвалы радка `s`
`Insert (s1,s,index);`	Працэдура ўстаўляе падрадок `s1` у радок `s`

Ва ўсіх камандах пераменная `s` абазначае зыходны радок, над якім выконваецца аперацыя. Пераменная `index` абазначае пазіцыю сімвала, пачынаючы з якога выконваюць аперацыю, а пераменная `count` — колькасць сімвалаў. Разбяром каманды падрабязней (прыклад 8.6).

Каманда `copy` з'яўляецца функцыяй, і вынік яе работы прысвойваецца іншай пераменнай. Радок `s` пры гэтым не змяняецца.

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

Каманды `delete` і `insert` з'яўляюцца працэдурамі, яны змяняюць радок `s`. У прыкладзе 8.7 паказана, як ужываюцца дадзеныя каманды.

Прыклад 8.6. Каманды для пераўтварэння радкоў.

1. Запіс `s1 := Copy(s, index, count);` азначае, што ў радку `s` вылучаюць `count` сімвалаў, вылучэнне пачынаюць з сімвала, індэкс якога запісаны ў пераменнай `index`. Гэтыя сімвалы капіруюцца ў пераменную `s1` (дзеянне каманды параўнальнае з капіраваннем фрагмента тэксту ў буфер абмену).

2. Запіс `Delete(s, index, count);` азначае, што ў радку `s` вылучаюць `count` сімвалаў, вылучэнне пачынаюць з сімвала, індэкс якога запісаны ў пераменнай `index`. Вылучаныя сімвалы выдаляюцца з радка `s`. Астатнія сімвалы радка зрушваюцца ўлева (дзеянне каманды параўнальнае з выдаленнем фрагмента тэксту).

3. Запіс `Insert(s1, s, index);` азначае, што ў радок `s` устаўляюць сімвалы радка `s1`, устаўка адбываецца ў пазіцыі `index`. Астатнія сімвалы радка зрушваюцца ўправа (дзеянне каманды параўнальнае з устаўкай фрагмента тэксту з буфера абмену).

Прыклад 8.7. Прыклады выкарыстання каманд.

`s := 'Інфарматыка';`	
`s1 := copy(s,3,5);`	`s1 = 'фарма'`
`Delete(s,8,4);`	`s = 'Інфарма'`
`Insert ('цыя',s,8);`	`s = 'Інфармацыя'`

Прыклад 8.8.

V. Праграма:

```
var  s, p, t: string;
     n1, n2, k: integer;
begin
  writeln('Радок s');
  readln(s);
  writeln('Падрадок 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('Сустракаецца ', k,
' раз(-ы,-оў)');
end.
```

VI. Тэсціраванне.

Запусціць праграму, увесці радок «Не слово хозяин хозяину, а хозяин слову хозяин» і падрадок «хозяин». Вынік:

Окно вывода

```
Радок s
Не слово хозяин хозяину, а хозяин слову хозяин
Падрадок p
хозяин
Сустракаецца 4 раз(-ы,-оў)
```

Калі для таго ж радка ўвесці падрадок «хозяйка», то вынік будзе такім:

Окно вывода

```
Радок s
Не слово хозяин хозяину, а хозяин слову хозяин
Падрадок p
хозяйка
Сустракаецца 0 раз(-ы,-оў)
```

Прыклад 8.8. Напісаць праграму, якая вызначыць, колькі разоў зададзены падрадок сустракаецца ў радку.

Этапы выканання задання

I. Зыходныя даныя: пераменная s — зыходны радок, p — зыходны падрадок.

II. Вынік: k — шуканая колькасць.

III. Алгарытм рашэння задачы.

1. Уводзім зыходныя даныя.

2. Ініцыялізуем значэнне лічыльніка k := 0;

3. Вызначаем даўжыні n1 і n2 для радка s і падрадка p.

4. У цыкле **for** ад 1 да розніцы ў даўжынях радка s і падрадка p:

4.1. Вылучаем з радка s падрадок t такой жа даўжыні, што і даўжыня p, пачынаючы з бягучага сімвала.

4.2. Параўноўваем падрадкі. Калі яны роўныя, то павялічваем значэнне лічыльніка на 1.

5. Выводзім вынік.

IV. Апісанне пераменных: s, p, t – string, n1, n2, k – integer.

Прыклад 8.9. Напісаць праграму, якая са слова ТЕСТИРОВАНИЕ атрымае слова РИСОВАНИЕ, выкарыстоўваючы працэдуры і функцыі пераўтварэння радкоў.

Этапы выканання задання

I. Зыходныя даныя: слова ТЕСТИРОВАНИЕ будзем захоўваць як канстанту з імем s.

II. Вынік: атрыманае слова.

III. Алгарытм рашэння задачы.

1. Вынік не залежыць ад даных, якія ўводзяцца.

2. У радок s1 запішам шосты сімвал зыходнага радка.

3. Скапіруем з радка s восем сімвалаў, пачынаючы з пазіцыі 5. Дабавім да радка s1. Атрымаем 'РИРОВАНИЕ'.

4. У атрыманым радку выдалім трэці сімвал ('РИОВАНИЕ').

5. Уставім на трэцяе месца трэці сімвал зыходнага радка ('РИСОВАНИЕ').

6. Выведзем вынік.

IV. Апісанне пераменных: s1 — string.

Прыклад 8.9.

V. Праграма:

```
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. Тэсціраванне.

Окно вывода

```
Слово = РИСОВАНИЕ
```

Для капіравання апошніх 8 сімвалаў з радка s можна выкарыстаць функцыю RightStr: s1 := s1 + RightStr(s,8).

Для капіравання першых сімвалаў з радка можна выкарыстаць функцыю LeftStr.

1. Што такое даўжыня радка?
2. З дапамогай якой функцыі можна знайсці даўжыню радка?
3. Якія функцыі выкарыстоўваюцца для вызначэння пазіцыі падрадка?
4. Як скапіраваць сімвалы з аднаго радка ў іншы?
5. Якая працэдура выкарыстоўваецца для выдалення сімвалаў з радка?
6. Якая працэдура прызначана для ўстаўкі сімвалаў у радок?
7*. Супастаўце каманды пераўтварэння радкоў з аперацыямі, якія выконваюцца з дапамогай буфера абмену.

Практыкаванні

1 У праграму з прыкладу 8.3 унеслі наступныя змяненні:

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

Як цяпер выводзіцца слова? Растлумачце чаму.

2 Змяніце праграму з прыкладу 8.3 так, як вызначана ніжэй.

1. Кожная літара павінна выводзіцца сваім колерам (можна выкарыстаць выпадковае заданне колераў).

2. Літары, якія стаяць на цотных месцах, павінны выводзіцца адным колерам, а на няцотных — іншым.

3 Змяніце праграму з прыкладу 8.4 так, каб на экран выводзіўся сімвал уведзенага слова, які стаіць пасярэдзіне (для слоў з цотнай колькасцю літар — сімвал справа ад сярэдзіны).

1. Праверце правільнасць работы сваёй праграмы на прапанаваных прыкладах. Адкрыйце файл з табліцай і запішыце вынікі.

Слова	Вынік
Школа	о
гімназія	а
форма	
Інтэрнэт	

2. Дапішыце ў табліцу два свае прыклады.

3. Што будзе выведзена, калі нічога не ўводзіць, проста націснуць Enter?

4. Праверце, ці сустракаецца выведзены сімвал у слове яшчэ раз.

5. Выведзіце пазіцыі ўсіх сімвалаў, што супадаюць з сімвалам слова, які знаходзіцца пасярэдзіне.

4 Змяніце праграму з прыкладу 8.4 так, каб малыя і вялікія літары аналізаваліся праграмай аднолькава (напрыклад, для слова «Алена» адказ павінен быць наступным: «`Апошні сімвал — а, сімвал сустрэўся на месцы 1`»).

5 Дадзены два словы. Ці правільна, што адно са слоў пачынаецца на тую ж літару, на якую заканчваецца другое? (Першая літара аднаго са слоў можа быць вялікай.) Калі так, то вывесці літару, інакш — адпаведнае паведамленне.

1. Праверце правільнасць работы сваёй праграмы на прапанаваных прыкладах. Адкрыйце файл з табліцай і запішыце вынікі.

Слова 1	Слова 2	Вынік
array	yellow	y
apple	auto	няправільна
Рыба	Акацыя	

2. Дапішыце ў табліцу два свае значэнні.

3*. Калі адказ «правільна», вызначыць, ці належаць літары аднаму рэгістру.

6 Змяніце праграму з прыкладу 8.8 так, каб пры `k = 0` выводзілася паведамленне `'Падрадок у радку не сустракаецца'`.

7 Праграму з прыкладу 8.8 запісалі наступным чынам:

```
var s, p, t: string;
  k, i: integer;
begin
 writeln('Радок s');
 readln(s);
 writeln('Падрадок 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('Сустракаецца', k, 'раз(-ы, -оў)');
end.
```

Параўнайце гэту праграму і праграму з прыкладу 8.8, вызначыўшы, колькі разоў выканаецца каманда цыкла ў кожнай з праграм для пералічаных выпадкаў.

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

*Ці можна падабраць тэст, для якога колькасць выкананняў цыкла будзе аднолькавай для абедзвюх праграм? Калі так, то які?

8 Атрымайце са слова «ТЕСТИРОВАНИЕ» дадзеныя словы. Для гэтага выкарыстоўвайце каманды: copy, delete, insert і аперацыю складання радкоў.

1. РОСТ.
2. НИВА.
3. ТОВАР.
4. ТОСТЕР.
5. ОТВЕРСТИЕ.
6. Прыдумайце сваё слова.

§ 9. Складанне алгарытмаў апрацоўкі радковых велічынь

9.1. Аналіз тэксту на наяўнасць розных сімвалаў

Сучасныя тэкставыя рэдактары дазваляюць атрымаць статыстыку па сімвалах і словах у дакуменце (прыклад 9.1).

Правільны набор тэксту прадугледжвае наяўнасць толькі аднаго прабелу паміж словамі. У правільна набраным тэксце колькасць слоў будзе на адзінку большая, чым колькасць прабелаў.

Прыклад 9.1. Статыстыка ў дакуменце Word (адпаведная каманда на ўкладцы **Рецензирование**).