Програмування на мові Паскаль Частина II

Тема 1. Масиви

© К.Ю. Поляков

Переклад: Р. М. Васильчик

Масиви

Масив – це група однотипних елементів, які мають спільне ім’я і розміщені в пам’яті поряд.

Особливості:

- всі елементи мають **один тип**
- весь масив має **одне ім’я**
- всі елементи розміщені в пам’яті **поряд**

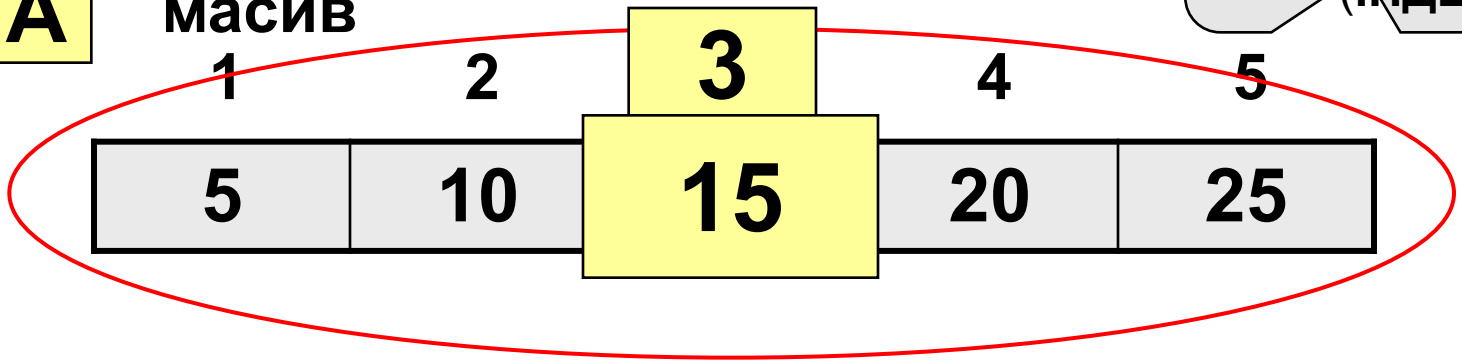
Приклади:

- список учнів в класі
- квартири в будинку
- школи в місті
- дані про температуру повітря за рік

Масиви

НОМЕР
елемента масиву
(**ІНДЕКС**)

A масив



A[1] A[2] A[5]

ЗНАЧЕННЯ
елемента масиву

НОМЕР (ІНДЕКС)
елемента масиву: 2

A[2]

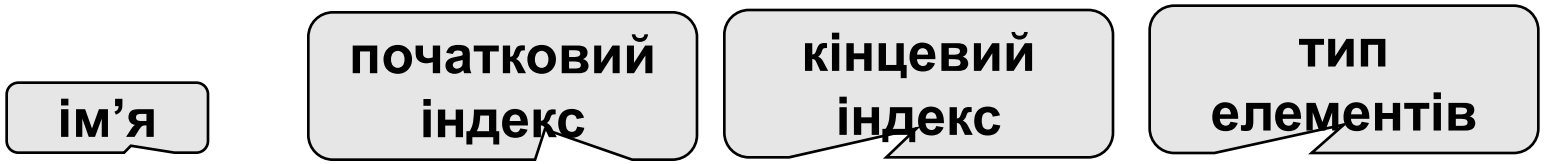
ЗНАЧЕННЯ
елемента масиву: 10

Оголошення масивів

Для чого оголошувати?

- визначити **ім'я** масиву
- визначити **тип** масиву
- визначити **кількість елементів**
- виділити **місце в пам'яті**

Масив цілих чисел:




```
var A: array[ 1 .. 5 ] of integer ;
```

Розмір через константу:

```
const N=5;
var A: array[1..N] of integer;
```

Оголошення масивів

Масиви інших типів:

```
var X, Y: array [1..10] of real;
    C: array [1..20] of char;
```

Інший діапазон індексів:

```
var Q: array [0..9] of real;
    C: array [-5..13] of char;
```

Інд

```
var A: array ['A'..'Z'] of real;
    B: array [False..True] of integer;
...
   A['C'] := 3.14259*A['B'];
   B[False] := B[False] + 1;
```

Що неправильно?

```
var a: array [1..10] of integer;
...
  A[5] := 4.5;
```

```
var a: array ['a'..'z'] of integer;
...
  A['b'] := 15;
```

```
var a: array [0..9] of integer;
...
  A[10] := 'x';
```

Масиви

Оголошення:

```
const N = 5;
var a: array[1..N] of integer;
    i: integer;
```

Введення з клавіатури:

```
for i:=1 to N do begin
  write('a[', i, ']=');
  read ( a[i] );
end;
```

```
a[1] = 5
a[2] = 12
a[3] = 34
a[4] = 56
a[5] = 13
```

Чому write?

Ви

```
for i:=1 to N do a[i]:=a[i]*2;
```

```
writeln('Масив A:');
for i:=1 to N do
  write(a[i]:4);
```

```
Масив A:
  10  24  68 112  26
```

Завдання

"4": Ввести з клавіатури масив з 5 елементів, знайти середнє арифметичне всіх елементів масиву.

Приклад:

```
Введіть п’ять чисел:
4   15   3  10    14
середнє арифметичне 9.200
```

"5": Ввести з клавіатури масив з 5 елементів, знайти мінімальний з них.

Приклад:

```
Введіть п’ять чисел:
4   15    3    10    14
мінімальний елемент 3
```

При зміні N решта програми не повинна змінюватися!

Програмування на мові Паскаль
Частина II

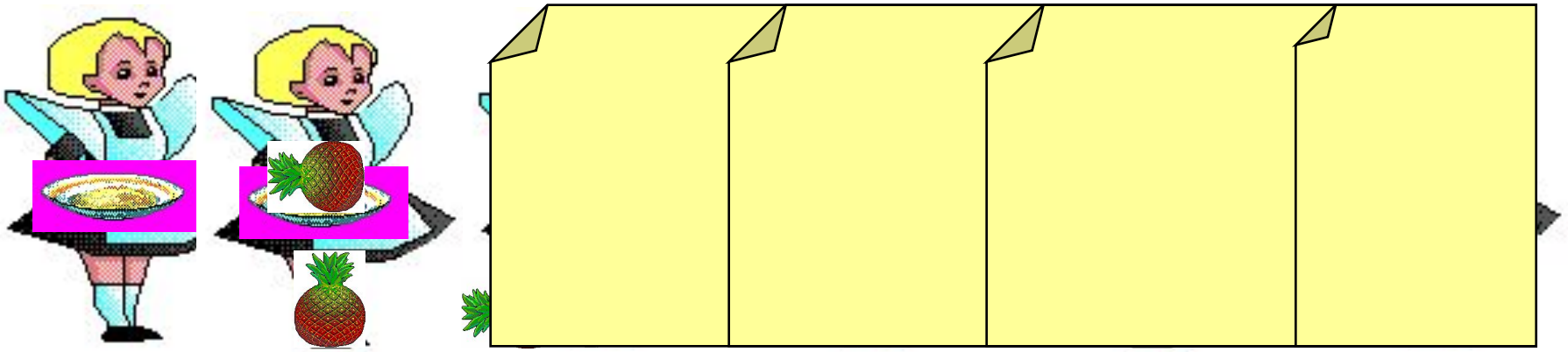
Тема 2. Максимальний елемент масиву

Переклад: Р. М. Васильчик

Максимальний елемент

Задача: знайти в масиві максимальний елемент.

Алгоритм:



Псевдокод:

```
{ вважаємо, що перший елемент – максимальний }
for i:=2 to N do
  if a[i] > { максимального } then
    { запам'ятати новий максимальний елемент a[i] }
```

Чому цикл від i=2?

Максимальний елемент

Додатково: як знайти номер максимального елемента?

```
                  { вважаємо, що перший – максимальний }
iMax := 1;
for i:=2 to N do        { перевіряємо всі решта }
  if a[i] > a[iMax] then { знайшли новий максимальний }
  begin
                        { запам'ятати a[i] }
    iMax := i;          { запам'ятати i }
  end;
```

? Як спростити?

По номеру елемента **iMax** завжди можна знайти його значення **a[iMax]**. Тому всюди замінюємо max на **a[iMax]** і забираємо змінну **max**.

Програма

```
program qq;
const N = 5;
var a: array [1..N] of integer;
    i, iMax: integer;
begin
  writeln('Вихідний масив:');
  for i:=1 to N do begin
    a[i] := random(100) + 50;
    write(a[i]:4);
  end;
  iMax := 1; { вважаємо, що перший – максимальний }
  for i:=2 to N do      { перевіряємо всі решта }
    if a[i] > a[iMax] then { новий максимальний }
      iMax := i;           { запам'ятати і }
  writeln; {перейти на новий рядок}
  writeln('Максимальний елемент a[', iMax, ']=', a[iMax]);
end;
```

випадкові числа в інтервалі [50,150)

пошук максимального

Завдання

"4": Заповнити масив з 10 елементів випадковими числами з інтервалу [-10..10] і знайти в ньому максимальний і мінімальний елементи та їх номери.

Приклад:

```
Вихідний масив:
4   -5   3  10  -4  -6  8  -10  1  0
максимальний a[4]=10
мінімальний  a[8]=-10
```

"5": Заповнити масив з 10 елементів випадковими числами з інтервалу [-10..10] і знайти в ньому два максимальних елементи та їх номери.

Пример:

```
Вихідний масив:
4   -5   3  10  -4  -6  8  -10  1  0
максимальний a[4]=10, a[7]=8
```

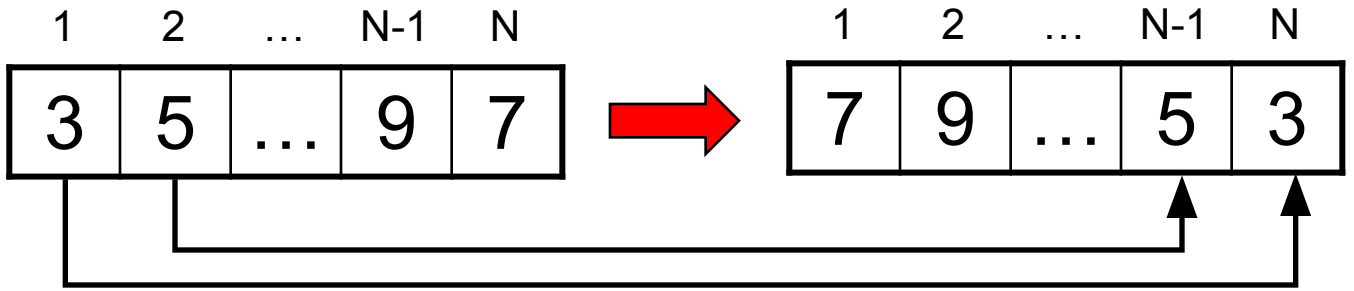
Програмування на мові Паскаль
Частина II

Тема 3. Опрацювання масивів

© К.Ю. Поляков
Переклад: Р. М. Васильчик

Інверсія масиву

Задача: переставити елементи масиву в зворотному порядку.



Алгоритм:

сума індексів N+1

поміняти місцями **A[1]** і **A[N]**, **A[2]** і **A[N-1]**, **…**

Псевдокод:

```
for i:=1 to N div 2 do
  { поміняти місцями A[i] і A[N+1-i] }
```

Що неправильно?

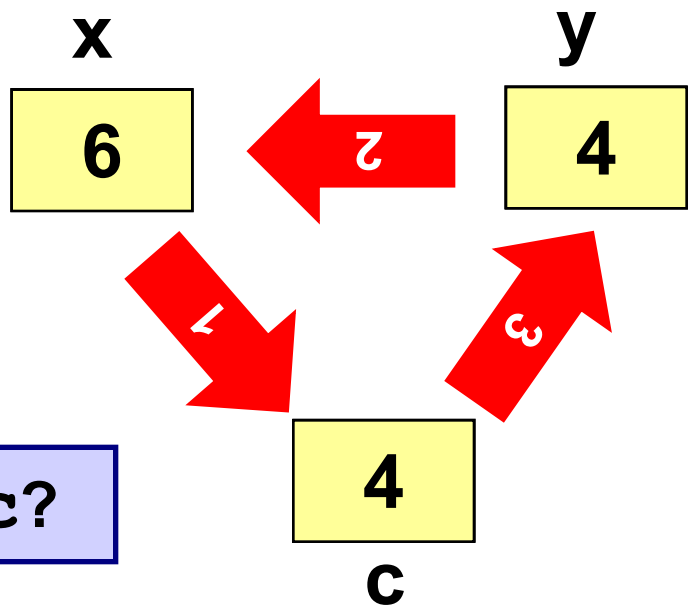
Як переставити елементи?

Задача: поміняти місцями вміст двох чашок.

Задача: поміняти місцями вміст двох комірок пам'яті.

```
x := y;
y := x;
```

```
c := x;
x := y;
y := c;
```



Чи можна обійтися без c?

Програма

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, c: integer;
begin
  { заповнити масив }
  { вивести вихідний масив }
  for i:=1 to N div 2 do begin
    c:=A[i]; A[i]:=A[N+1-i]; A[N+1-i]:=c;
  end;
  { вивести одержаний масив }
end;
```

Завдання

"4": Заповнити масив з 10 елементів випадковими числами з інтервалу [-10..10] і виконати інверсію окремо для 1-ї і 2-ї половини масиву.

Приклад:

```
Вихідний масив:
4   -5   3  10  -4 | -6  8  -10  1  0
Результат:
-4  10   3  -5   4 |  0  1  -10  8 -6
```

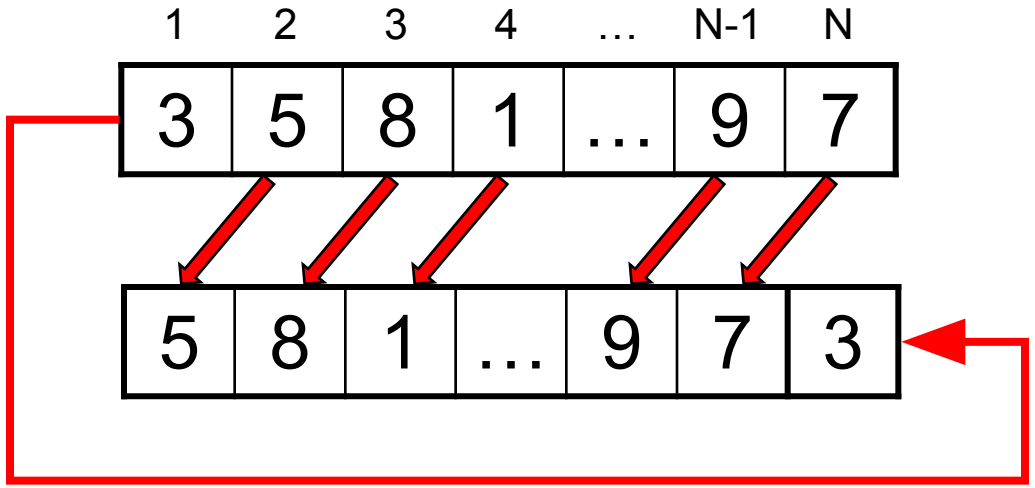
"5": Заповнити масив з 12 елементів випадковими числами з інтервалу [-12..12] і виконати інверсію для кожної третини масиву.

Приклад:

```
Вихідний масив:
4   -5   3  10 | -4  -6   8 -10 | 1  0  5  7
Результат:
10   3  -5   4 |-10   8  -6  -4 | 7  5  0  1
```

Циклічний зсув

Задача: зсунути елементи масиву на 1 комірку, перший елемент стає на місце останнього.



Алгоритм:

```
A[1]:=A[2]; A[2]:=A[3];… A[N-1]:=A[N];
```

Цикл:

чому не **N**?

```
for i:=1 to N-1 do
  A[i]:=A[i+1];
```

Що неправильно?

Програма

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, c: integer;
begin
  { заповнити масив }
  { вивести вихідний масив }
  c := A[1];
  for i:=1 to N-1 do A[i]:=A[i+1];
  A[N] := c;
  { вивести одержаний масив }
end;
```

Завдання

"4": Заповнити масив з 10 елементів випадковими числами з інтервалу [-10..10] і виконати циклічний зсув ВПРАВО.

Приклад:

```
Вихідний масив:
4   -5   3  10  -4  -6  8  -10  1  0
Результат:
 0   4   -5   3  10  -4  -6  8  -10  1
```

"5": Заповнити масив з 12 елементів випадковими числами з інтервалу [-12..12] і виконати циклічний зсув ВПРАВО на 4 елементи.

Приклад:

```
Вихідний масив:
4   -5   3  10  -4  -6   8 -10  1  0  5  7
Результат:
-4  -6   8 -10   1   0   5   7  4 -5  3  10
```

Програмування на мові Паскаль
Частина II

Тема 4. Сортування масивів

© К.Ю. Поляков
Переклад: Р. М. Васильчик

Сортування

Сортування – це розстановка елементів масиву в заданому порядку (по зростанню, спаданню, останній цифрі, сумі дільників, …).

Задача: переставити елементи масиву в порядку зростання.

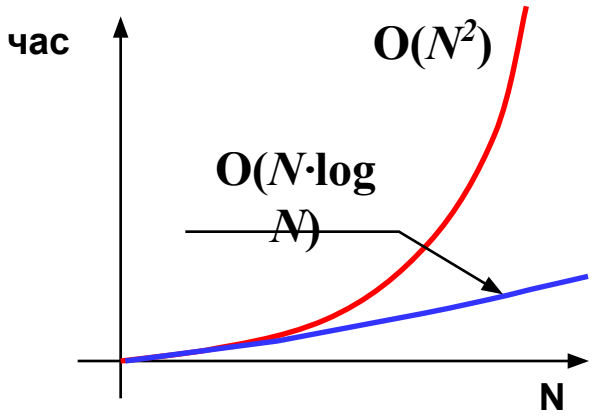
Алгоритми:

складність $O(N^2)$

- прості і зрозумілі, проте неефективні для переважної більшості масивів
  - **метод бульбашки**
  - **метод вибору**

складність $O(N \cdot \log N)$

- складні, проте ефективні
  - **“швидке сортування" (*Quick Sort*)**
  - **сортування “купою" (*Heap Sort*)**
  - **сортування злиттям**
  - **пірамідальне сортування**

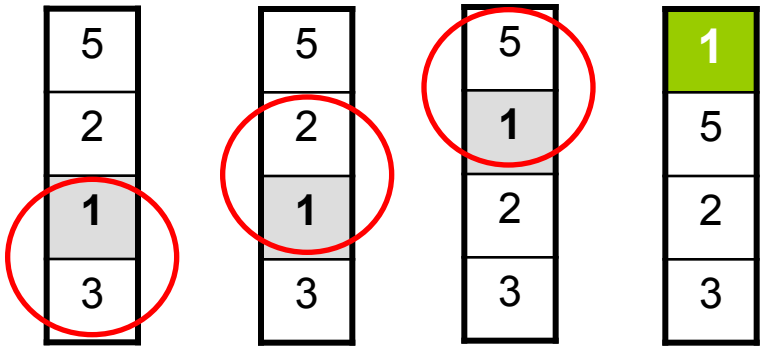


Метод бульбашки

Ідея – бульбашка повітря в стакані води піднімається з дна вверх.

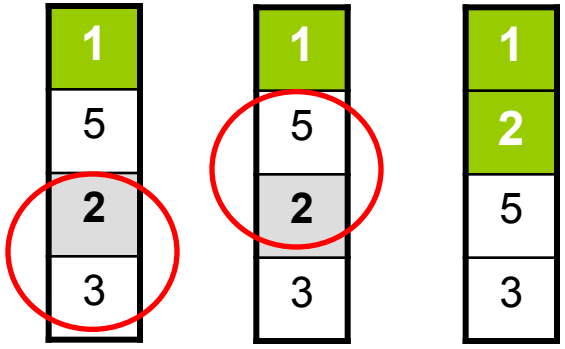
Для масивів – самий маленький ("легкий") елемент переміщується вверх ("спливає").

1-ий прохід

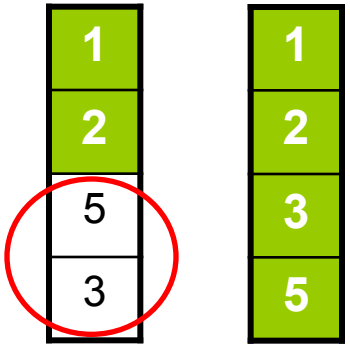


- починаємо знизу, порівнюємо два сусідніх елементи; вони стоять “неправильно”, міняємо їх місцями
- за 1 прохід по масиву **один** елемент (самий маленький) стає на своє місце

2-ий прохід



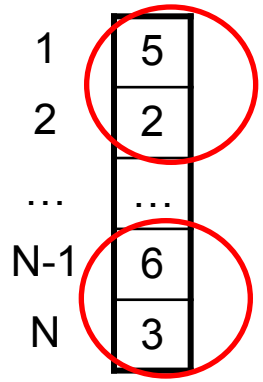
3-ій прохід



Для сортування масиву з N елементів потрібен N-1 прохід (достатньо поставить на свої місця N-1 елемент).

Програма

1-ий прохід:



порівнюються пари

`A[N-1] і A[N],  A[N-2] і A[N-1]`

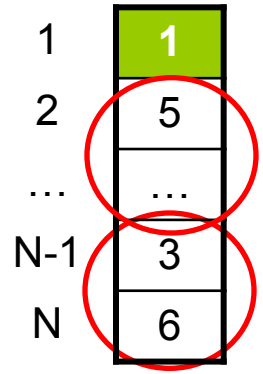
…

`A[1] і A[2]`

`A[j] і A[j+1]`

```
for j:=N-1 downto 1 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

2-ий прохід



A[1] вже на своєму місці!

```
for j:=N-1 downto 2 do
  if A[j] > A[j+1] then begin
    c:=A[j]; A[j]:=A[j+1]; A[j+1]:=c;
  end;
```

i-ий прохід

```
for j:=N-1 downto i do
  ...
```

Програма

```
program qq;
const N = 10;
var A: array[1..N] of integer;
    i, j, c: integer;
begin
  { заповнити масив }
  { вивести вихідний масив }
  for i:=1 to N-1 do begin
    for j:=N-1 downto i do
      if A[j] > A[j+1] then begin
        c := A[j];
        A[j] := A[j+1];
        A[j+1] := c;
      end;
   end;
  { вивести одержаний масив }
end;
```

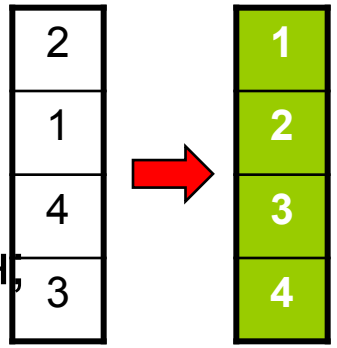
Чому цикл по i до N-1?

елементи вище A[i] вже поставлені

Метод бульбашки з прапором

Ідея – якщо при виконанні методу бульбашки не було обмінів, масив вже посортований і решта проходів не потрібні.

Реалізація: **змінна-прапор**, показує, був чи ні обмін; якщо вона дорівнює **`False`**, то вихід.




```
var flag: boolean;
```

```
repeat
  flag := False; { скинути прапор }
  for j:=N-1 downto 1 do
    if A[j] > A[j+1] then begin
      c := A[j];
      A[j] := A[j+1];
      A[j+1] := c;
      flag := True; { підняти прапор }
    end;
until not flag;  { вихід при flag=False }
```

? **Як покращити?**

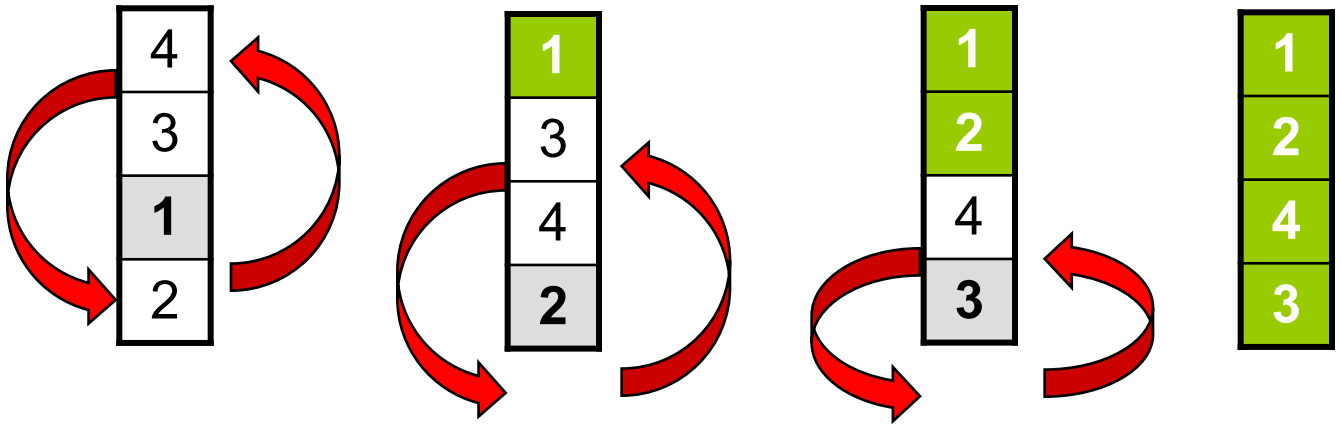
Метод бульбашки з прапором

```
  i := 0;
repeat
  i := i + 1;
  flag := False; { скинути прапор }
  for j:=N-1 downto i do
    if A[j] > A[j+1] then begin
      c := A[j];
      A[j] := A[j+1];
      A[j+1] := c;
      flag := True; { підняти прапор }
    end;
until not flag;  { вихід при flag=False }
```

Метод вибору

Ідея:

- знайти мінімальний елемент і поставити на місце першого (помінять місцями з `A[1]`)
- **із решти** знайти мінімальний елемент і поставити на друге місце (поміняти місцями з `A[2]`), і т.д.



Метод вибору

потрібен N-1 прохід

```
for i := 1 to N-1 do begin
  nMin = i;
  for j:= i+1 to N do
    if A[j] < A[nMin] then nMin:=j;
  if nMin <> i then begin
    c:=A[i];
    A[i]:=A[nMin];
    A[nMin]:=c;
  end;
end;
```

пошук мінімального від A[i] до A[N]

якщо потрібно, переставляємо

? Чи можна забрати if?

Завдання

"4": Заповнити масив з 10 елементів випадковими числами з інтервалу [0..100] і відсортувати його за останньою цифрою.

Приклад:

```
Вихідний масив:
14  25  13  30  76  58  32  11  41  97
Результат:
30  11  41  32  13  14  25  76  97  58
```

"5": Заповнити масив з 10 елементів випадковими числами з інтервалу [0..100] і відсортувати першу половину по зростанню, а другу – по спаданню.

Приклад:

```
Вихідний масив:
14  25  13  30  76 | 58  32  11  41  97
Результат:
13  14  25  30  76 | 97  58  41  32  11
```

Програмування на мові Паскаль
Частина II

Тема 5. Пошук в масиві

© К.Ю. Поляков
Переклад: Р. М. Васильчик

Пошук в масиві

Задача – знайти в масиві елемент, рівний **X**, або встановити, що його немає.

Розв'язання: для довільного масиву: **лінійний пошук** (перебір)

недостаток: **низька швидкість**

Як спростити? – завчасно підготувати масив для пошуку

- як саме підготувати?
- як використовувати "підготовлений масив"?

Лінійний пошук

nX – номер потрібного елемента в масиві

```
nX := 0; { поки не знайшли ...}
for i:=1 to N do { цикл по всіх елементах }
  if A[i] = X then { якщо знайшли, то ... }
     nX := i;      { ... запам'ятати номер}
if nX < 1 then writeln('Не нашли...')
else           writeln('A[', nX, ']=', X);
```

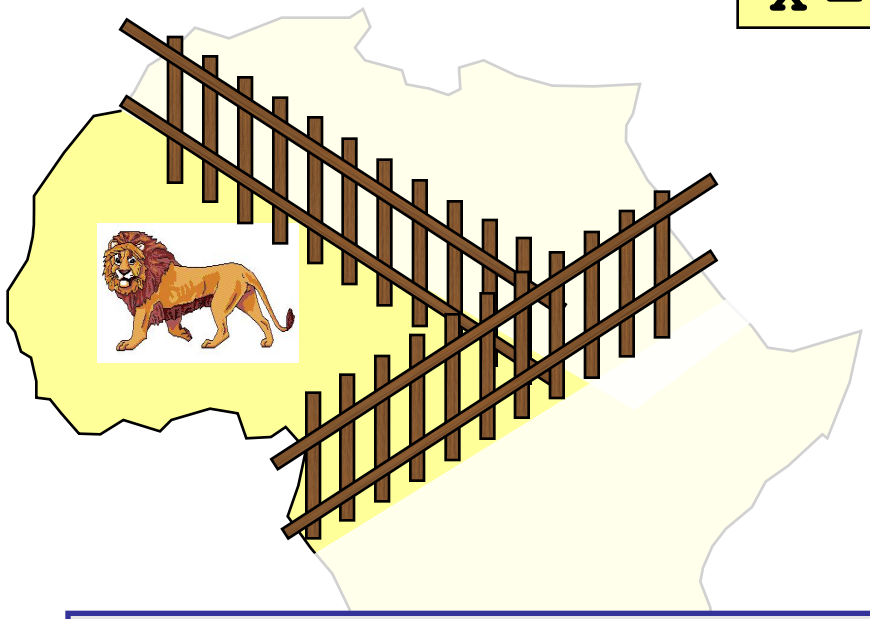
Покращення: після того, як знайшли **X**, виходимо з циклу.

Що погано?

```
nX := 0;
for i:=1 to N do
 if A[i] = X then begin
   nX := i;
   break ;{вихід з циклу}
 end;
```

```
nX := 0; i := 1;
while i <= N do begin
  if A[i] = X then begin
    nX := i; i := N;
  end;
  i := i + 1;
end;
```

Двійковий пошук



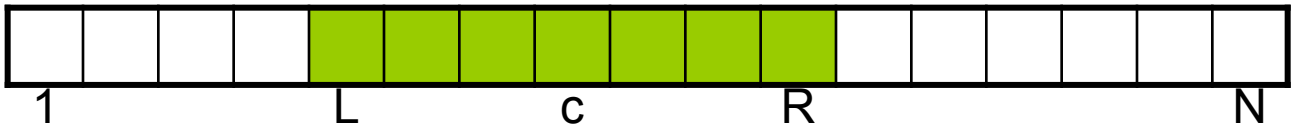
X = 7

1	1	1	1
2	2	2	2
3	3	3	3
4		**4**	4
5		5	5
6		6	**6**
7		7	7
8		8	8
9		9	9
10		10	10
11		11	11
12		12	12
13		13	13
14		14	14
15		15	15
16		16	16

X < 8 ↑ **X > 4** ↓ **X > 6** ↓

1. **Вибрати середній елемент `A[c]` і порівняти з X.**
2. **Якщо `X = A[c]`, знайшли (вихід).**
3. **Якщо `X < A[c]`, шукати дальше в першій половині.**
4. **Якщо `X > A[c]`, шукати дальше в другій половині.**

Двійковий пошук




```
nX := 0;
L := 1; R := N; {межі: шукаємо від A[1] до A[N] }
while R >= L do begin
  c := (R + L) div 2;
  if X = A[c] then begin
     nX := c;
     R := L - 1; { break; }
  end;
  if x < A[c] then R := c - 1;
  if x > A[c] then L := c + 1;
end;
if nX < 1 then writeln('Не знайшли...')
else           writeln('A[', nX, ']=', X);
```

номер середнього елемента

знайшли

вийти з циклу

зсуваємо межі

Чому не можна `while R > L do begin … end;` ?

Порівняння методів пошуку

	Лінійний	Двійковий
підготовка	ні	відсортувати
	кількість кроків	
N = 2	2	2
N = 16	16	5
N = 1024	1024	11
N= 1048576	1048576	21
N	$\leq N$	$\leq \log_2 N+1$

Завдання

"4": **Написати програму, яка сортує масив ПО СПАДАННЮ і шукає в ньому елемент, рівний X (це число вводиться з клавіатури). Використати двійковий пошук.**

"5": **Написати програму, яка рахує середню кількість кроків в двійковому пошуку для масиву з 32 елементів з інтервалу [0,100]. Для пошуку використати 1000 випадкових чисел в цьому ж інтервалі.**

Програмування на мові Паскаль
Частина II

Тема 6. Символьні рядки

Переклад: Р. М. Васильчик

Чим поганий масив символів?

Це масив символів:

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

- кожен символ – окремий об’єкт;
- масив має довжину N, яка задана при оголошенні

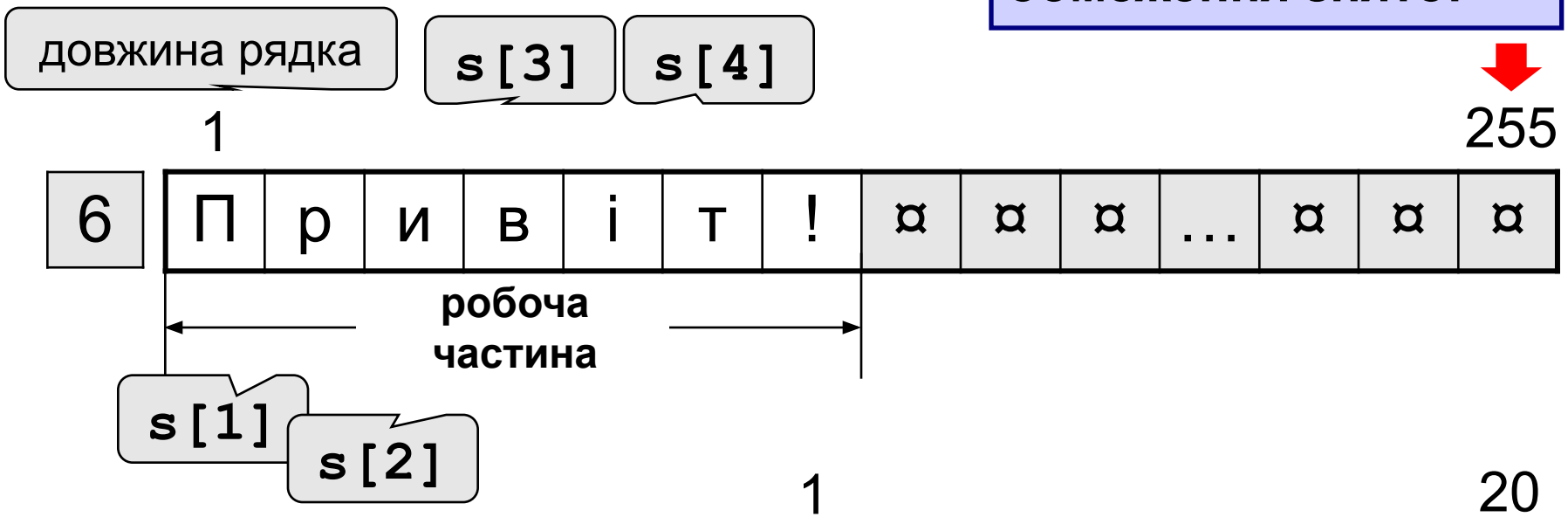
Що потрібно:

- опрацьовувати послідовність символів як єдине ціле
- рядок повинен мати змінну довжину

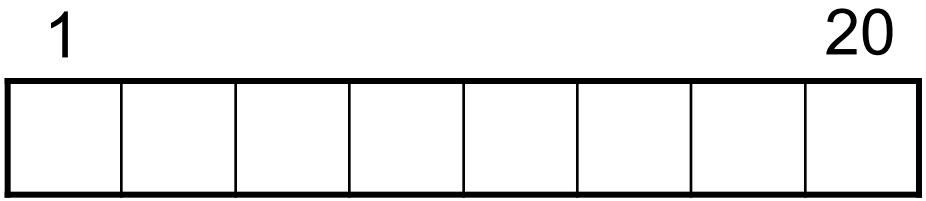
Символьні рядки

```
var s: string;
```

В *Delphi* це обмеження знято!




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



Довжина рядка:

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

```
var i: integer;
```

Символьні рядки

Задача: ввести рядок з клавіатури і замінити всі букви "а" на букви "б".

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

введення рядка

довжина рядка

виведення рядка

Завдання

"4": **Ввести символьний рядок і замінити всі букви "а" на букви "б" і навпаки, як великі, так і маленькі.**

Приклад:

```
Ввести рядок:
ааббссААББСС
Результат:
ббаассББААСС
```

"5": **Ввести символьний рядок і перевірити, чи є він паліндромом (паліндром читається однаково в обох напрямках).**

Приклад:

```
Введіть рядок:
АБВГДЕ
Результат:
Не паліндром.
```

Приклад:

```
Введіть рядок:
КОРОК
Результат:
Паліндром.
```

Операції з рядками

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

Запис нового значення:

```
s := 'Вася';
```

Об'єднання: додати один рядок в кінець другого.

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

`'Привіт, Вася!'`

Підрядок: повернути частину рядка з іншого рядка.

```
s := '123456789';

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

з 3-го символу

6 штук

`'345678'`

`'456'`

Знищення і вставка

Знищення частини рядка:

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

6 штук

'123456789'

'129'

рядок міняється!

з 3-го символу

Вставка в рядок:

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

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

починаючи з 3-го символу

'12ABC3456789'

що вставляємо

куди вставляємо

'12ABQC3456789'

Пошук в рядку

Пошук в рядку:

s[3]

var n: integer;

```
s := 'Тут був Вася.';
n := Pos ( 'у', s );
if n > 0 then
     writeln('Буква у - це s[', n, ']')
else writeln('Не знайшли');
n := Pos ( 'Вася', s );
s1 := Copy ( s, n, 4 );
```

3

n = 11

Особливості:

- функція повертає номер символу, з якого починається зразок в рядку
- якщо слова немає, повертається 0
- пошук з початку (знаходиться **перше** слово)

Приклади

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

```
6
'Вася  Мітя'
'Вася Катя Мітя'
```

```
s := 'Вася Петя Мітя';
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'
'Мітя'
'Петя'
'ПетяВасяМітя'
12
```

Приклад розв’язання задачі

Задача: Ввести ім’я, по батькові і прізвище. Перетворити їх до формату “прізвище-ініціали".

Приклад:

```
Введіть ім’я, по батькові і прізвище:
Василь Алібабаєвич Хрюндіков
Результат:
Хрюндіков В.А.
```

Алгоритм:

- знайти перший пропуск і виділити ім’я
- знищити ім’я з пропуском із основного рядка
- знайти перший пропуск і виділити по батькові
- знищити по батькові з пропуском із основного рядка
- “склеїти" прізвище, перші букви імені і фамілії, крапки, пропуски…

Програма

```
program qq;
var s, name, otch: string;
    n: integer;
begin
   writeln('Введіть ім'я, по батькові і прізвище');
   readln(s);
   n := Pos(' ', s);
   name := Copy(s, 1, n-1); { вирізати ім'я }
   Delete(s, 1, n);
   n := Pos(' ', s);
   otch := Copy(s, 1, n-1); { вирізати по батькові }
   Delete(s, 1, n);          { залишилось прізвище }
   s := s + ' ' + name[1] + '.' + otch[1] + '.';
   writeln(s);
end.
```

Завдання

"4": **Ввести ім'я файлу (можливо, без розширення) і змінити його розширення на ".exe".**

Приклад:

```
Ввести ім'я файлу:        Ввести ім'я файлу:
qqq                       qqq.com
Результат:                Результат:
qqq.exe                   qqq.exe
```

"5": **Ввести шлях до файлу і "розібрати" його, виводячи кожну вкладену папку з нового рядка**

Приклад:

```
Ввести шлях до файлу:
C:\Мої документи\10-Б\Вася\qq.exe
Результат:
C:
Мої документи
10-Б
Вася
qq.exe
```

Програмування на мові Паскаль
Частина II

Тема 7. Рекурсивний перебір

© К.Ю. Поляков
Переклад: Р. М. Васильчик

Рекурсивний перебір

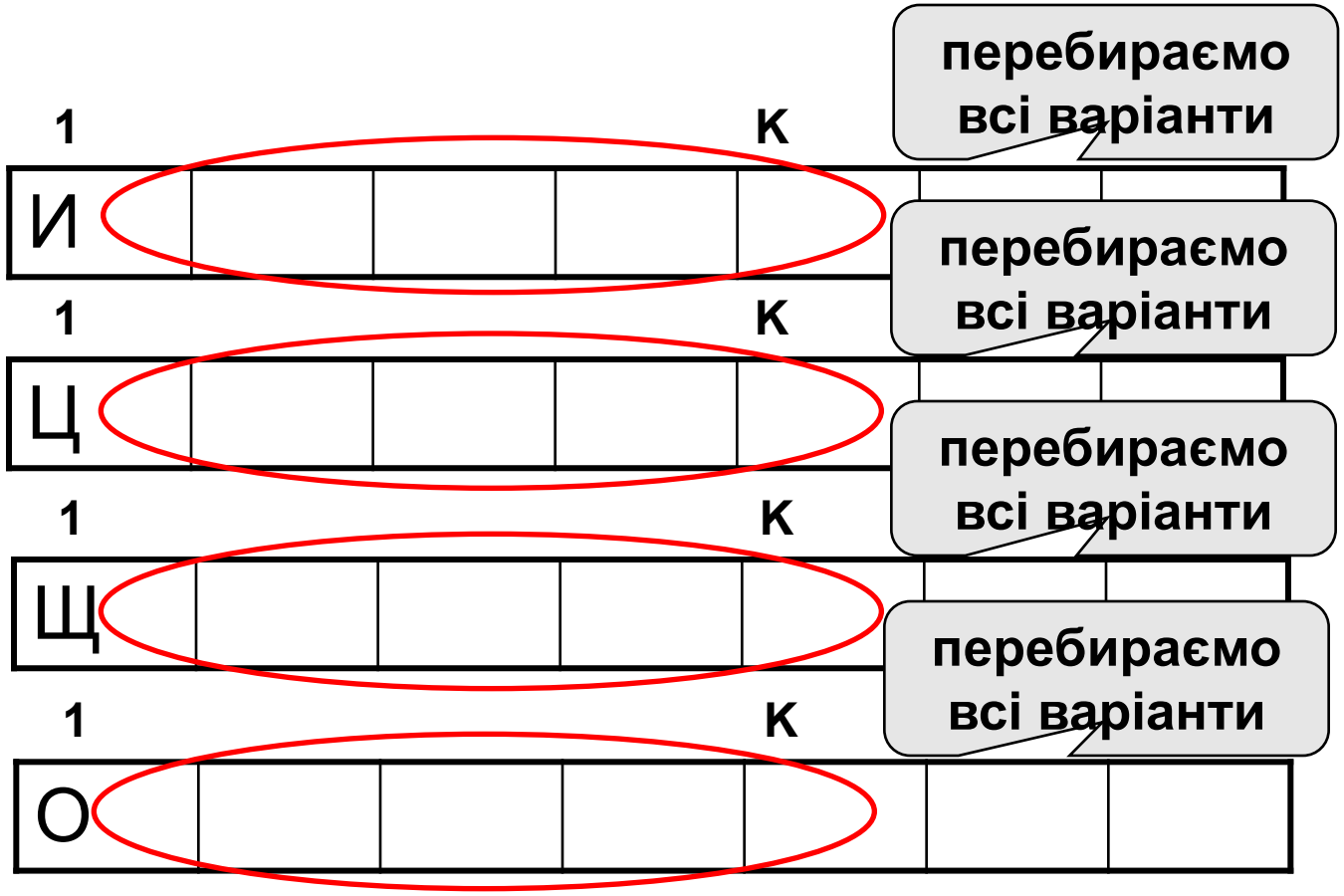
Задача: Алфавіт мови племені "тумба-юмба" складається з букв **И**, **Ц**, **Щ** і **О**. Вивести на екран всі слова із **К** букв, які можна скласти в цій мові, і підрахувати їх кількість. Число **К** вводиться з клавіатури.

Кількість варіантів:

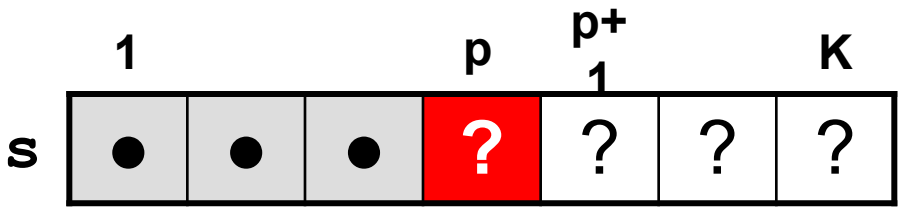
$$N = 4 \cdot 4 \cdot 4 \cdot \boxtimes \cdot 4 = 4^K$$

Рекурсивний перебір

Рекурсія: Розв'язання задачі для слів з **К** букв зводиться до 4-х задач для слів з **К-1** букви.



Процедура



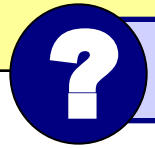

```
procedure Rec(p: integer);
begin
  if p > K then begin
    writeln(s);
    count := count+1;
  end
  else begin
    s[p]:='Ы'; Rec ( p+1 );
    s[p]:='Ц'; Rec ( p+1 );
    s[p]:='Щ'; Rec ( p+1 );
    s[p]:='О'; Rec ( p+1 );
  end;
end;
```

Глобальні змінні:

```
var s: string;
    count, K: integer;
```

закінчення рекурсії

рекурсивні виклики



А якщо букв багато?

Процедура

```
procedure Rec(p: integer);
const letters = 'ИЦЩО';
var i: integer;
begin
  if p > k then begin
     writeln(s);
     count := count+1;
  end
  else begin
     for i:=1 to length(letters) do begin
       s[p] := letters[i];
       Rec(p+1);
     end;
  end;
end;
```

всі букви

локальна змінна

цикл по всіх буквах

Програма

```
program qq;
var s: string;
    K, i, count: integer;
    procedure Rec(p: integer);
    ...
    end;
begin
  writeln('Введіть довжину слова:');
  read ( K );
  s := '';
  for i:=1 to K do s := s + ' ';
  Rec ( 1 );
  writeln('Всього ', count, ' слів');
end.
```

глобальні змінні

процедура

рядок з K пропусків

Завдання

Алфавіт мови племені "тумба-юмба" складається з букв **И**, **Ц**, **Щ** і **О**. Число **К** вводиться з клавіатури.

"4": Вивести на екран всі слова з **К** букв, в яких буква **И** зустрічається більше 1 разу, і підрахувати їх кількість.

"5": Вивести на екран всі слова з **К** букв, в яких є однакові букви, що стоять поряд (наприклад, **ИЩЩО**), і підрахувати їх кількість.

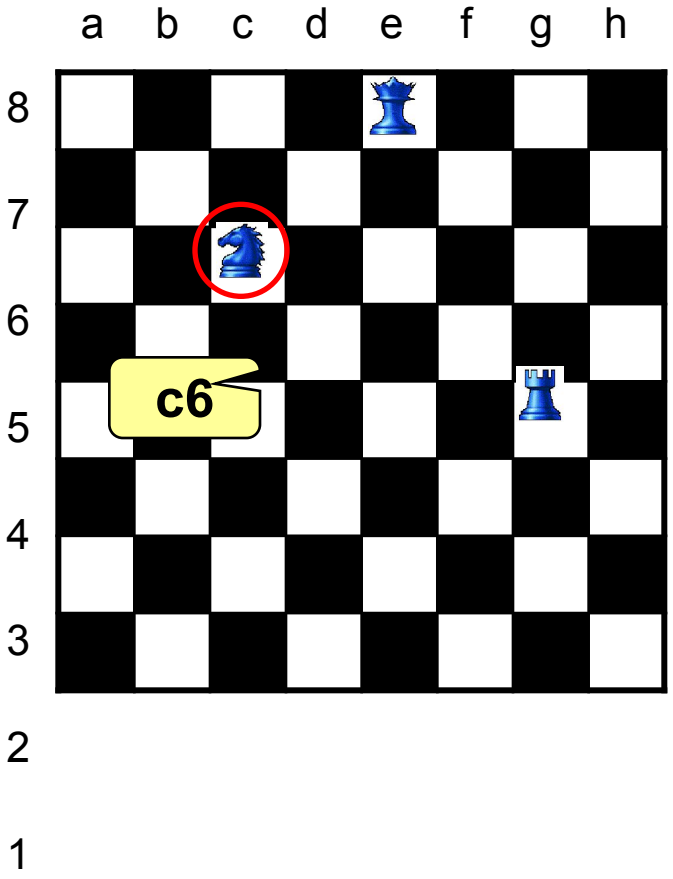
Програмування на мові Паскаль
Частина II

Тема 8. Матриці

Переклад: Р. М. Васильчик

Матриці

Задача: запам’ятати розміщення фігур на шаховій дошці.



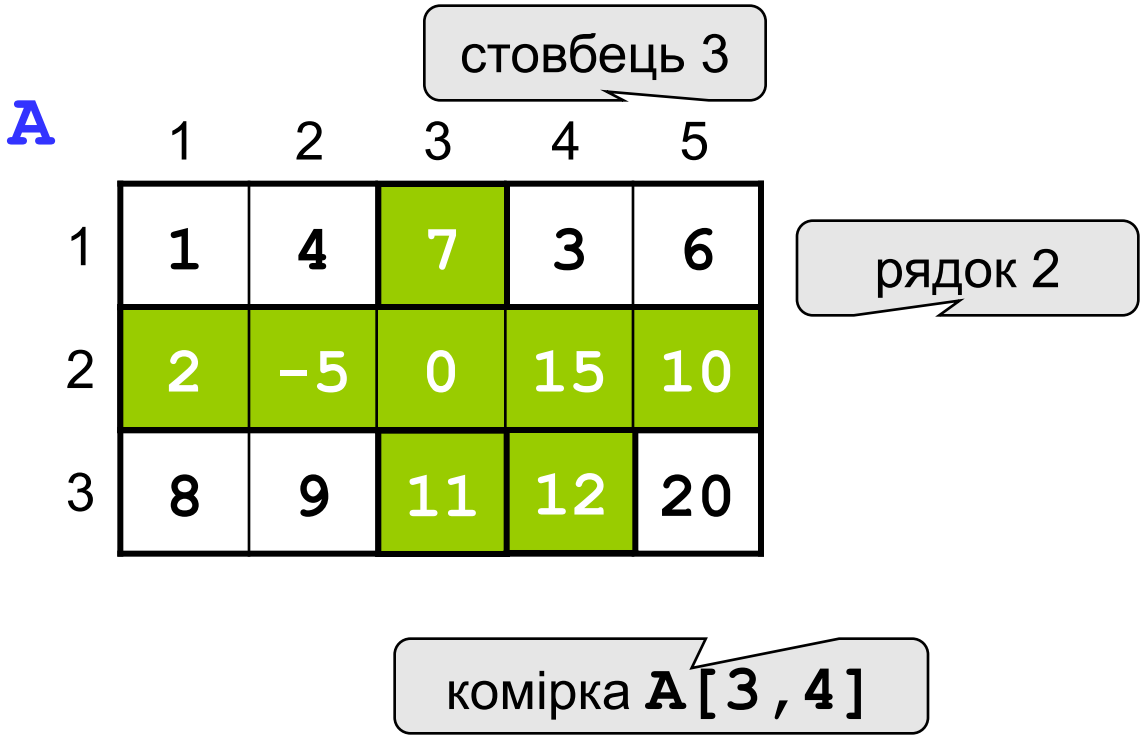
	1	2	3	4	5	6	7	8
8	0	0	0	0	2	0	0	0
7	0	0	0	0	0	0	0	0
6	0	0	3	0	0	0	0	0
5	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	4	0
3	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

A[6,3]

Матриці

Матриця – це прямокутна таблиця чисел.

Матриця – це масив, в якому кожний елемент має два індекси (номер рядка і номер стовпця).



Матриці

Оголошення:

```
const N = 3;
      M = 4;
var A: array[1..N,1..M] of integer;
    B: array[-3..0,-8..M] of integer;
    Q: array['a'..'d',False..True] of real;
```

Введення з клавіатури:

Якщо переставити цикли?

```
for j:=1 to M do
  for i:=1 to N do begin
    write('A[',i,',',j,']=');
    read ( A[i,j] );
  end;
```

i j

A[1,1] = 2
A[1,2] = 5
A[1,3] = 1
...
A[3,4] = 4

Матриці

Заповнення випадковими числами

```
for i:=1 to N do
  for j:=1 to M do
    A[i,j] := random(25) - 10;
```

цикл по рядках

цикл по стовпцях

ервал?

Вывод на экран

```
for i:=1 to N do begin
  for j:=1 to M do
    write ( A[i,j]:5 );
  writeln;
end;
```

виведення рядка

в тому ж рядку

перейти на новий рядок

```
   12   25    1   13
   15              44
    6    1   12     7
    1   45   22
        6    2    23
```

Якщо переставити цикли?

Опрацювання всіх елементів матриці

Задача: заповнити матрицю з 3 рядків і 4 стовпців випадковими числами і вивести її на екран. Знайти суму елементів матриці.

```
program qq;
const N = 3; M = 4;
var A: array[1..N,1..M] of integer;
    i, j, S: integer;
begin
  ... { заповнення матриці і виведення на екран}
  S := 0;
  for i:=1 to N do
    for j:=1 to M do
      S := S + A[i,j];
  writeln('Сума елементів матриці ', S);
end;
```

Завдання

Заповнити матрицю з 8 рядків і 5 стовпців випадковими числами з інтервалу [-10,10] і вивести її на екран.

"4": Знайти мінімальний і максимальний елемент в матриці і їх номера. Формат виведення:

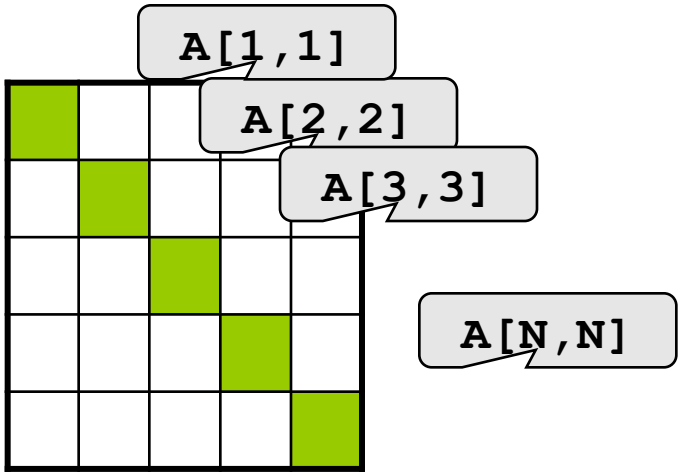
```
Мінімальний елемент  A[3,4]=-6
Максимальний елемент A[2,2]=10
```

"5": Вивести на екран рядок, сума елементів якого максимальна. Формат виведення:

```
Рядок 2:  3  5  8  9  8
```

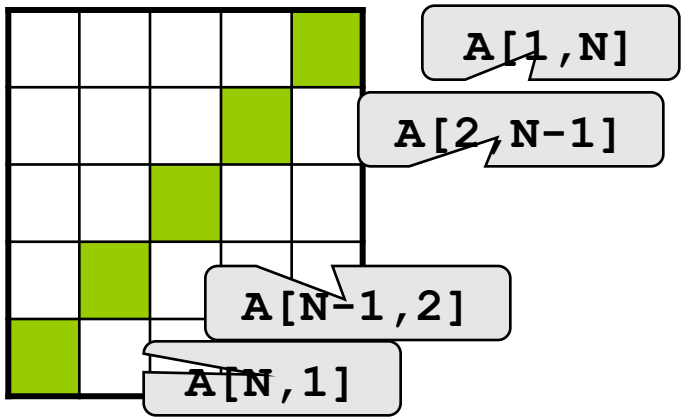
Операції з матрицями

Задача 1. Вивести на екран головну діагональ квадратної матриці з N рядків і N стовпців.




```
for i:=1 to N do
  write ( A[i,i]:5 );
```

Задача 2. Вивести на екран другу діагональ.

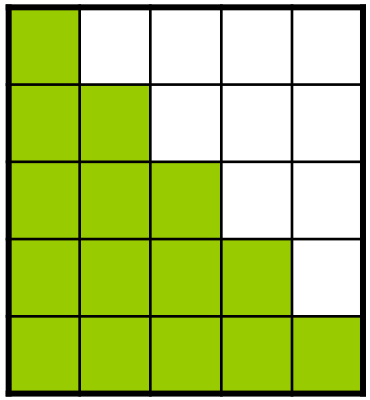


сума номерів рядка в стовпця N+1

```
for i:=1 to N do
  write ( A[i, N+1-i ]:5 );
```

Операції з матрицями

Задача 3. Знайти суму елементів, які стоять на головній діагоналі і нижче її.



Одиночний цикл чи вкладений?

рядок 1: `A[1,1]`

рядок 2: `A[2,1]+A[2,2]`

`...`

рядок N: `A[N,1]+A[N,2]+...+A[N,N]`

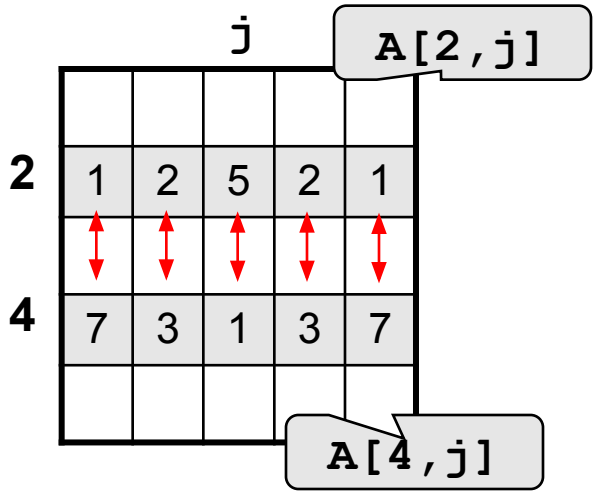
```
S := 0;
for i:=1 to N do
  for j:=1 to i do
    S := S + A[i,j];
```

цикл по всіх рядках

додаємо потрібні елементи рядка **i**

Операції з матрицями

Задача 4. Перестановка рядків або стовпців. В матриці з N рядків і M стовпців переставити 2-й і 4-й рядок.




```
for j:=1 to M do begin
  c := A[2,j];
  A[2,j] := A[4,j];
  A[4,j] := c;
end;
```

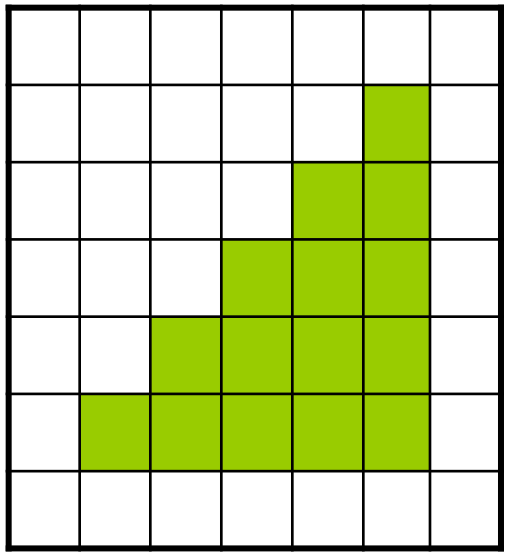
Задача 5. До третього стовпця додати шостий.

```
for i:=1 to N do
  A[i,3] := A[i,3] + A[i,6];
```

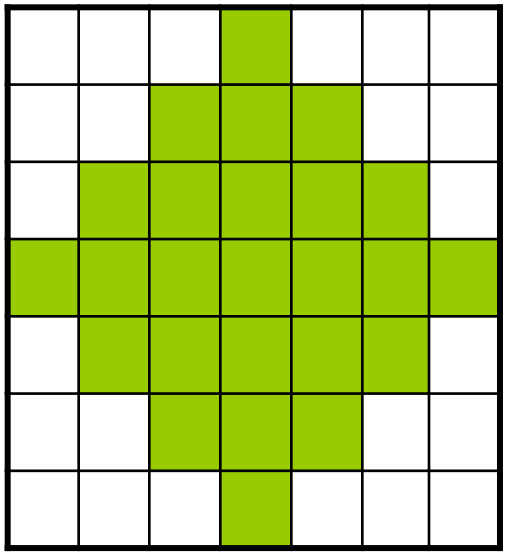
Завдання

Заповнити матрицю з 7 рядків і 7 стовпців випадковими числами з інтервалу [-10,10] і вивести її на екран. Обнулити елементи, відмічені зеленим фоном, і вивести одержану матрицю на екран.

"4":



"5":



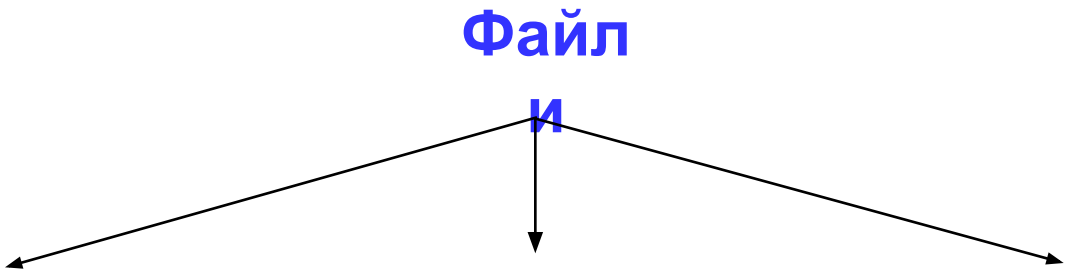
Програмування на мові Паскаль
Частина II

Тема 9. Файли

© К.Ю. Поляков
Переклад: Р. М. Васильчик

Файли

Файл – це область на диску, яка має ім’я.



Текстові

тільки текст без оформлення, не містить керівних символів (з кодами < 32)

ACSII (1 байт на символ)

UNICODE (2 байта на символ)

`*.txt, *.log,`
`*.htm, *.html`

Двійкові

може містити будь-які символи кодової таблиці

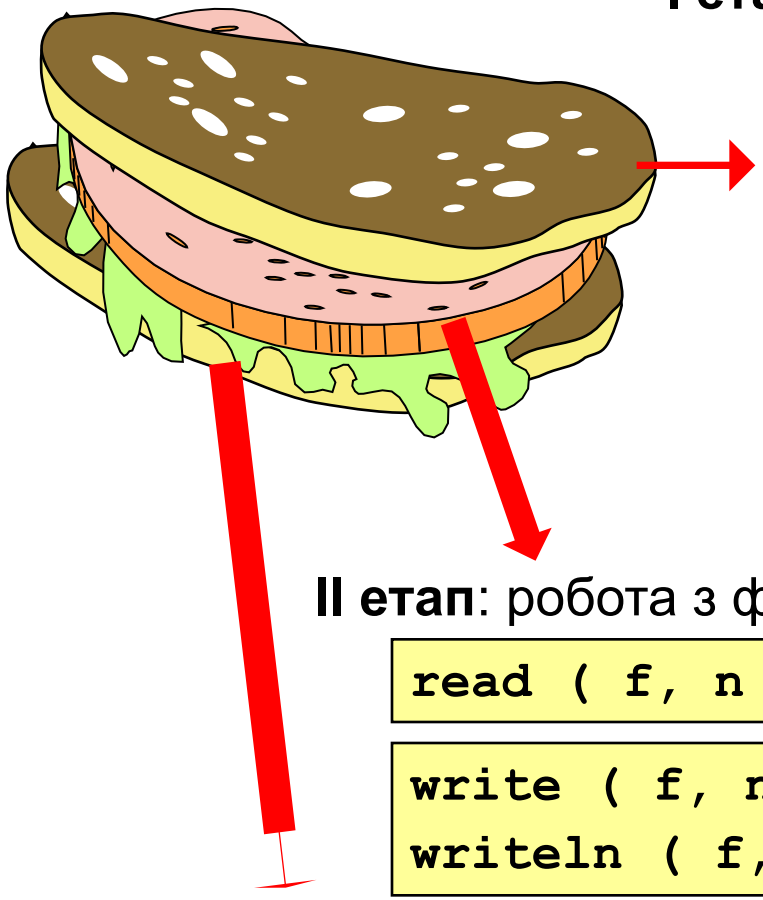
`*.doc, *.exe,`
`*.bmp, *.jpg,`
`*.wav, *.mp3,`
`*.avi, *.mpg`

Папки (каталоги)

Принцип сендвіча

Змінна типу «текстовий файл":

```
var f: text;
```



I етап. відкрити файл :

- зв'язати змінну **f** з файлом

```
assign(f, 'qq.dat');
```

- відкрити файл (зробити його активним, приготувати до роботи)

```
reset(f); {для читання}
```

```
rewrite(f); {для запису}
```

II етап: робота з файлом

```
read ( f, n );  { ввести значення n }
```

```
write ( f, n );  { записати значення n }
writeln ( f, n );{з переходом на новий рядок}
```

III етап: закрити файл

```
close(f);
```

Робота з файлами

Особливості:

- ім’я файлу згадується тільки в команді **`assign`**, звернення до файлу іде через файлову змінну
- файл, який відкривається для читання, повинен **існувати**
- якщо файл, який відкривається на запис, існує, то старий вміст **знищується**
- дані записуються в файл у текстовому вигляді
- при завершенні програми всі файли закриваються автоматично
- після закриття файлу змінну **`f`** можна використовувати ще раз для роботи з іншим файлом

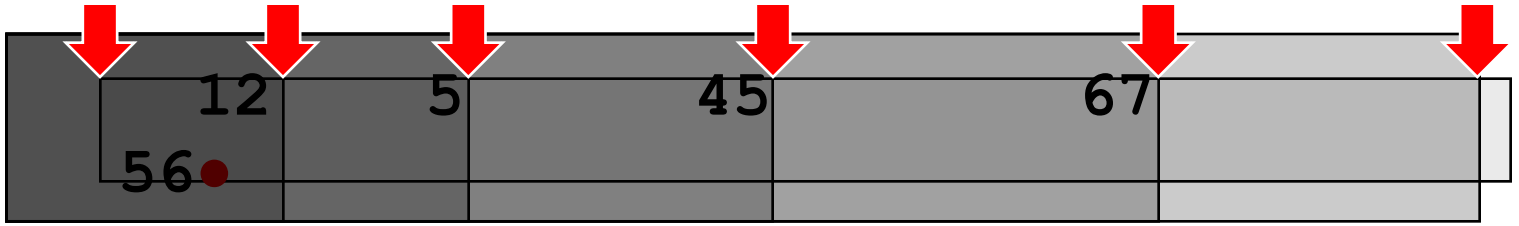
Послідовний доступ

- при відкритті файлу курсор встановлюється в початок

```
assign ( f, 'qq.dat' );
reset ( f );
```

- читання виконується з тієї позиції, де стоїть курсор
- після читання курсор зміщується на перший непрочитаний символ

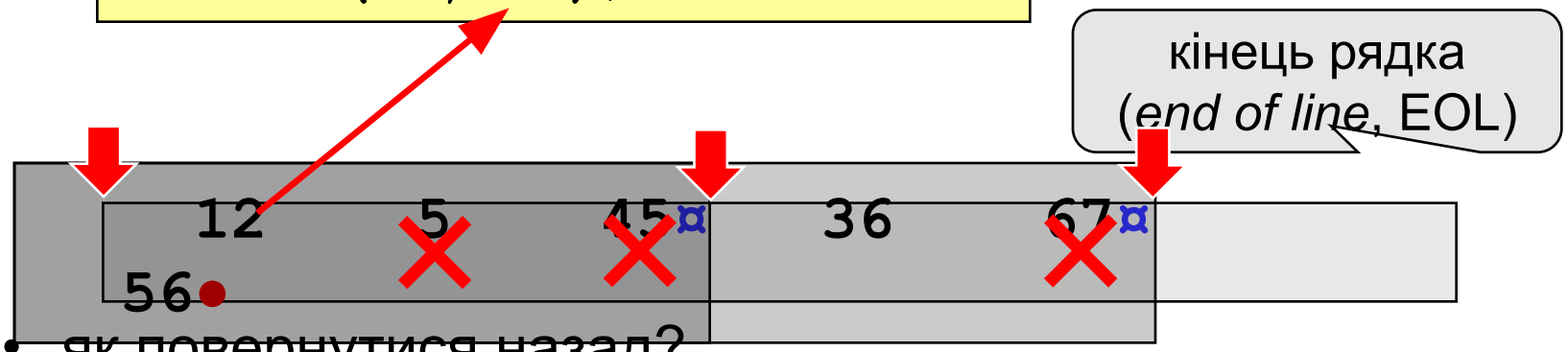
```
read ( f, x );
```



Послідовний доступ

- читання до кінця рядка

```
readln ( f, x );
```



- як повернутися назад?

```
close ( f );
reset ( f ); { почати з початку }
```

Приклад

Задача: **в файлі `input.txt` записані числа (в стовпчик), скільки їх – невідомо. Записати в файл `output.txt` їх суму.**

Чи можна обійтися без масиву?

Алгоритм:

1. Відкрити файл **`input.txt`** для читання.
2. **`S := 0;`**
3. Якщо чисел не залишилося, перейти до кроку 7.
4. Прочитати наступне число в змінну **`x`**.
5. **`S := S + x;`**
6. Перейти до кроку 3.
7. Закрити файл **`input.txt`**.
8. Відкрити файл **`output.txt`** для запису.
9. Записати в файл значення **`S`**.
10. Закрити файл **`output.txt`**.

цикл з умовою **«поки є дані"**

Програма

```
program qq;
var s, x: integer;
    f: text;
begin
   assign(f, 'input.txt');
   reset(f);
   s := 0;
   while not eof(f) do begin
      readln(f, x);
      s := s + x;
   end;
   close(f);
   assign(f, 'output.txt');
   rewrite(f);
   writeln(f, 'Сума чисел ', s);
   close(f);
end.
```

логічна функція, повертає **True**, якщо досягнуто кінець файлу

запис результату у файл **output.txt**

Завдання

В файлі `input.txt` записані числа, скільки їх – невідомо.

"4": Знайти середнє арифметичне всіх чисел і записати його в файл **`output.txt`**.

"5": Знайти мінімальне і максимальне число і записати їх в файл **`output.txt`**.

Опрацювання масивів

Задача: в файлі `input.txt` записані числа (в стовпчик), скільки їх – невідомо, але не більше 100. Переставити їх в порядку зростання і записати в файл `output.txt`.

Чи можна обійтися без масиву?

Проблеми:

1. для сортування потрібно утримувати в пам'яті всі числа одночасно (масив);
2. скільки чисел – невідомо.

Розв'язання:

3. виділяємо в пам'яті масив з 100 елементів;
4. записуємо прочитані числа в масив і рахуємо їх в змінній `N`;
5. сортуємо перші `N` елементів масиву;
6. записуємо їх в файл.

Читання даних в масив

Глобальні змінні:

```
var A: array[1..100] of integer;
    f: text;
```

Функція: вводить масив, повертає кількість елементів

```
function ReadArray: integer;
var i: integer;
begin
  assign(f, 'input.txt');
  reset(f);
  i := 0;
  while (not eof(f)) and (i < 100) do begin
    i := i + 1;
    readln(f, A[i]);
  end;
  close(f);
  ReadArray := i;
end;
```

цикл закінчується, якщо досягнутий кінець файлу або прочитано 100 чисел

Програма

```
program qq;
var A: array[1..100] of integer;
    f: text;
    N: integer;
    function ReadArray: integer;
    ...
    end;
Begi

   N := ReadArray;
   ... { сортування перших N елементів }
   assign(f, 'output.dat');
   rewrite(f);
   for i:=1 to N do
     writeln(f, A[i]);
   close(f);
end.
```

вивід відсортованого масиву у файл

Завдання

В файлі `input.txt` записані числа (в стовпчик), відомо, що їх не більше 100.

"4": Відсортувати масив по спаданню останньої цифри і записати його в файл **`output.txt`**.

"5": Відсортувати масив по зростанню суми цифр і записати його в файл **`output.txt`**.

Опрацювання текстових даних

Задача: в файлі `input.txt` записані рядки, в яких є слово-паразит "*коротше*". Очистити текст від мусора і записати в файл `output.txt`.

Файл `input.txt` :

```
Мама, коротше, мила, коротше, раму.
Декан, коротше, пропив, коротше, бутан.
А роза, коротше, упала на лапу, коротше, Азора.
Кожний, коротше, мисливець бажає, коротше, знати, де
   ...
```

Результат - файл `output.txt` :

```
Мама мила раму.
Декан пропив бутан.
А роза упала на лапу Азора.
Кожний мисливець бажає знати, де сидить фазан.
```

Обробка текстових даних

поки не закінчилися дані

Алгоритм:

1. Прочитати рядок з файлу (`readln`).
2. Знищити всі слова "*, коротше,*" (`Pos, Delete`).
3. Перейти до кроку 1.

Опрацювання рядка `s`:

шукати ", коротше,"

знищити 9 символів

```
repeat
  i := Pos(', коротше,', s);
  if i <> 0 then Delete(s, i, 9);
until i = 0;
```

Особливості:

потрібно одночасно тримати відкритими два файли (один в режимі читання, другий – в режимі запису).

Робота з файлами

```
program qq;
var s: string;
    i: integer;
    fIn, fOut:
       text;
begin
  assign(fIn, 'instr.txt');
  reset(fIn);
  assign(fOut, 'outstr.txt');
  rewrite(fOut);
  ... { опрацювати файл }
  close(fIn);
  close(fOut);
end.
```

файлові змінні

відкрити файл для читання

відкрити файл для запису

Повний цикл опрацювання файлів

поки не досягнутий кінець файла

```
while not eof(fIn) do begin
  readln(fIn, s);
  repeat
    i := Pos(', коротше,', s);
    if i <> 0 then
       Delete(s, i, 9);
  until i = 0;
end;
```

опрацювання рядка

запис "очищеного" рядка

Завдання

В файлі `input.txt` записані рядки, скільки їх – невідомо.

"4": Замінити всі слова "коротше" на "в загальному" і записати результат у файл **`output.txt`**.

"5": Вивести в файл **`output.txt`** тільки ті рядки, в яких більше 5 слів (слова розділені одним пропуском).

Кінець фільму