МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

М Е Т О Д И Ч Н І В К А З І В К И

до лабораторних і контрольних робіт

по програмуванню мовою Паскаль

ч а с т и н а 2

(для студентів спеціальності 6.091501
«Комп'ютерні мережі і системи»
прискореної форми навчання)

Укладачі доц.Назаренко В.И.,
ас.Дємєш Н.С.,
ас.Юсупова К.Б.

Розглянуто
на засіданні кафедри КІ
протокол № 1 від 31 серпня 2010 р.

Затверджено
на засіданні навчально-
видавничої ради ДонНТУ
протокол № 5 від 06.12.10

Донецьк ДонНТУ 2010

Навчальне видання

УДК 681.3(07)

Методичні вказівки до лабораторних і контрольних робіт по програмуванню мовою Паскаль, частина 2 (для студентів спеціальності 7.091501 “Комп'ютерні мережі і системи” прискореної форми навчання). Укладачі: В.И. Назаренко, Н.С.Дємєш, К.Б.Юсупова. - Донецьк: ДонНТУ, 2010. - 66 с.

Наведено методичні вказівки до лабораторної роботи № 2 і контрольної роботи № 2, що виконуються у другому семестрі відповідно до навчального плану спеціальності «Комп'ютерні мережі і системи» прискореної форми навчання. У лабораторній роботі № 2 «Використання процедур і функцій» розглядаються методи організації процедур і функцій, а також способи їхнього використання при обробці одновимірних масивів з довільними іменами типів. Змістом контрольної роботи № 2 «Обробка тексту» є методи організації програм у завданнях обробки тексту, представленого у вигляді масиву рядків. По кожній з контрольних робіт наведено 50 варіантів завдань.

Укладачі: доц. Назаренко В.І.,

ас. Дємєш Н.С.,

ас. Юсупова К.Б.

Відповідальний за випуск проф. Святний В.А.

Рецензент доц. Федяєв О.І.

Лабораторна робота №2

ВИКОРИСТАННЯ ПРОЦЕДУР І ФУНКЦІЙ

Методичні вказівки

Мета роботи - практично освоїти методи організації процедур і функцій, а також способи їхнього використання при обробці одновимірних масивів з різними іменами типів.

У кожному з наведених нижче завдань необхідно скласти програму, що включає в себе принаймні одну підпрограму (процедуру або функцію), що виконує основну обробку вхідних даних відповідно до умови завдання. Всі завдання сформульовані щодо обробки одного одновимірного масиву або пари одночасно оброблюваних масивів. У програмі, що реалізує лабораторну роботу № 2, повинна бути передбачена в першому випадку послідовна обробка **трьох різних масивів**, а в другому випадку - **трьох різних пар масивів**. При цьому для забезпечення універсальності розробленої підпрограми передбачається, що послідовно оброблювані в ній **масиви або пари масивів мають різні імена типів.**

Лабораторну роботу № 2 рекомендується виконувати в три етапи. Однак у пояснювальній записці потрібно приводити матеріали лише останнього етапу.

Етап 1. Програма складається в припущенні, що в ній обробляється тільки один масив (або одна пари масивів). На цьому етапі повинна бути ретельно перевірена правильність реалізації алгоритму рішення завдання шляхом прогону системи тестових вхідних даних. У систему тестів повинні бути включені не тільки "нормальні" масиви, але й масиви, які мають деякі граничні властивості (наприклад, масив, що складається з одних нулів, або масив, що складається з одного елемента). Для реалізації однотипних фрагментів програми рекомендується застосовувати процедури й функції; у складі програми доцільно застосовувати також інструментальні процедури й функції, які можуть бути використані практично без зміни в різних програмах (наприклад, вивід одновимірного масиву, контроль розміру сторінки, що виводиться на екран і т.п.).

Етап 2. Розроблена програма переробляється в процедуру або функцію в припущенні, що всі послідовно оброблювані масиви (або пари масивів) мають однакові імена типів. В основній програмі в цьому випадку виконується уведення й вивід вхідних даних, послідовне звертання до підпрограми, вивід результатів. Кожний з оброблюваних масивів повинен уводитися з окремого текстового файлу.

Етап 3. Виконується перетворення основної програми й підпрограм для загального випадку, що передбачає обробку масивів з різними іменами типів.

У деяких завданнях у лабораторній роботі № 3 потрібно виконувати пошук серій в одномірному масиві (серія - це група послідовно розташованих однотипних елементів). Розглянемо методику такого пошуку на конкретному прикладі.

Приклад. У цілочисельному масиві визначити положення найбільш довгої серії від'ємних елементів, причому до складу серії повинні входити не менш трьох елементів.

Нехай нам заданий масив $X = (x_1, x_2, ..., x_n)$. Тоді ознака початку серії - це істинність відношення $x_i < 0$, а ознака кінця серії - істинність відношення $x_i \geq 0$.

У програмі `GroupSearche` для шуканої серії ставляться у відповідність дві змінні: змінна `k1`, що визначає позицію першого елемента серії, і змінна `k2`, що відповідає кінцю серії (позиції елемента, що розташований після останнього елемента серії). У цьому випадку довжина серії (кількість елементів, які входять до її складу)

дорівнює k2-k1. Пошук початку чергової серії виконується з позиції k2+1, де k2 визначає кінцеву границю попередньої серії. Пошук кінця поточної серії здійснюється, починаючи з позиції k1+1, де k1 визначає початок даної серії.

У програмі використовуються функції SignBegin і SignEnd, які виконують пошук початку й кінця серії по відповідним ознакам. Якщо при звертанні до функції пошуку необхідна ознака не знайдена, то вихідне значення функції буде дорівнювати нулю.

Загальний пошук здійснюється в циклі **While** під керуванням булівської змінної Cond, що має значення true доти, поки триває пошук.

На старті загального пошуку змінній k2 привласнюється нульове значення, змінній Cond - значення true.

У початковій частині циклу **While** визначається положення чергової серії: k1:=SignBegin(k2+1). Якщо k1=0, то це означає, що в масиві таких серій більше немає. Тоді змінній Cond привласнюється значення false, що веде до припинення роботи циклу **While**. Якщо k1>0, то виконується пошук кінця серії: k2:=SignEnd(k1+1). Якщо k2=0, то це означає, що останній елемент масиву належить шуканій серії. Тоді змінній k2 привласнюється значення n+1, де n - кількість елементів у масиві, а змінній Cond - значення false. Після цього відповідно до умови завдання виконується обробка серії, обмеженої значеннями змінних k1 й k2.

```
Program GroupSearche;
Const  Nmax = 500;
Type   Ar = array[1..Nmax] of integer;
Var  i,k1,k2,
      l,           { розмір серії }
      lmax,        { розмір найбільш довгої серії }
      kmax         { позиція поч. елемента найбільш }
                   { довгої серії }
          : word;
      Cond : boolean; { керуюча змінна }
      X : Ar;         { оброблюваний масив }
{ ------------------------------------------------ }
Function SignBegin(k:word):word;
{ Пошук початку серії }
Var  i : word;
Begin
  SignBegin:=0;
  For i:=k to n do
    If x[i]<0 then
      Begin
        SignBegin:=i; Exit
      End;
End { SignBegin };
{ ------------------------------------------------ }
Function SignEnd(k:word):word;
{ Пошук кінця серії }
Var  i : word;
Begin
  SignEnd:=0;
  For i:=k to n do
    If x[i]>=0 then
      Begin
```

```
        SignEnd:=i; Exit
      End;
End { SignEnd };
{ ---------------------------------------------- }
Begin
  Уведення  n, X
  k2:=0; Cond:=true;
  lmax:=0; kmax:=0;
  While Cond do
    Begin
      k1:=SignBegin(k2+1);
      If k1=0 then
        Cond:=false
      Else
        Begin
          k2:=SignEnd(k1+1);
          If k2=0 then
            Begin
              k2:=n+1;
              Cond:=false
            End;
          l:=k2-k1;
          If (l>2) and (l>lmax) then
            Begin
              lmax:=l; kmax:=k1
            End;
        End;
    End;
 Вивід  lmax, kmax
End.
```

У деяких завданнях потрібно обчислювати значення різних поліномів. Такі обчислення повинні бути організовані за схемою Горнера, приклад програмної реалізації якої втримується в методичних вказівках до лабораторної роботи № 1. Зокрема, при перекладі числа із системи числення з підставою q у десяткову систему також потрібно застосовувати схему Горнера (окремо для цілої й дробової частин числа).

У пояснювальна записку по лабораторній роботі № 2 включаються ті ж пункти, що були наведені в методичних вказівках до лабораторної роботи №1, однак листинг програми повинен містити лише її варіант, отриманий на третьому етапі.

Приклад виконання завдання

Умова завдання.

На початку масиву X розмістити у вхідному відносному порядку всі додатні, після цього - всі нульові, а потім - всі від'ємні елементи даного масиву.

Етап 1. Виконаємо два варіанти рішення завдання: з використанням і без використання буферних масивів.

У програмі Cont3_1a використовуються два буферних масиви: у масив Xpos при перегляді вхідного масиву X записуються додатні елементи, у масив Xneg - від'ємні. Кількість елементів у масиві Xpos рівняється j, у масиві Xneg - k. Після цього в масив X пересилається вміст масиву Xpos, потім дописуються нульові елементи, кількість яких рівняється m = n - j - k, і на останньому етапі пересилається масив Xneg.

```
Program Cont3_1a;
Const Nmax=500;
Type Ar = array [1.. Nmax] of real;
Var  i,          { параметр циклу }
     j,          { кількість додатних елементів  }
     k,          { кількість від'ємних елементів }
     m,          { кількість нульових елементів }
     n : word;   { розмір вхідного масиву }
     X,          { вхідний масив }
     Xpos, Xneg: Ar; { буферні масиви }
Begin
        У в е д е н н я   n, X
  j:=0; k:=0;
  For i:=1 to n do
    If x[i]>0 then
      Begin
        Inc(j); Xpos[j]:=x[i];
      End
    Else
      If x[i]<0 then
        Begin
          Inc(k); Xneg[k]:=x[i]
        End;
  m:=n-j-k;
  For i:=1 to j do
    x[i]:=Xpos[i];
  For i:=j+1 to j+m do
    x[i]:=0;
  For i:=j+m+1 to n do
    x[i]:=Xneg[i-j-m];
  В и в і д    X
End.
```

Другий варіант завдання реалізований у програмі Cont3_1b. У зв'язку із забороною використання буферних масивів алгоритм її рішення істотно відрізняється від алгоритму попереднього варіанту.

На першому етапі роботи програми Cont3_1b здійснюється перестановка додатних елементів у початок масиву *X* зі збереженням їх вхідного відносного порядку. Перегляд елементів масиву *X* виконується в циклі **While**, починаючи з елемента з індексом *i* = 1. Якщо *i-ий* елемент додатний, то здійснюється перехід до наступного елемента *x*[*i*+1]; у противному випадку за допомогою функції SearchePos виконується пошук найближчого додатного елемента *x*[*j*], після чого здійснюється зрушення подмассива *x*[*i*].. *x*[*j*-1] вправо на одну позицію, а елементу *x*[*i*] привласнюється вхідне значення елемента *x*[*j*]. Цикл **While** працює під керуванням булівської змінної CondPos і припиняє свою роботу у двох випадках:

- у циклі проаналізований останній елемент із індексом *i* = *n*;

- починаючи з індексу *i*+1, у масиві не знайдені додатні елементи.

Якщо після закінчення циклу **While** має місце *i* > *n*, то це означає, що масив *X* містить тільки додатні елементи. У цьому випадку подальша обробка цього масиву не відбувається.

При *i* <= *n* з підмасива *x*[*i*+1] .. *x*[*n*] вилучаються нульові елементи, а їхня кількість формується в лічильнику CountZero. Елементи зміненого масиву, починаючи з позиції *i*+1, зрушують на CountZero позицій вправо, а в "звільнену" частину масиву дописуються нулі.

```
Program Cont3_1b;
Label  10;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  i,j,k,              { параметри циклів }
     n,                  { кількість елементів масиву X }
     CountZero : word;   { лічильник нульових елементів }
     R : integer;        { буферна змінна }
     CondPos : boolean;  { змінна керування циклом }
     X : Ar;             { вхідний масив }
   { ------------------------------------------------ }
Function SearchePos(k:word):word;
{ Пошук найближчого додатного елемента, починаючи }
{ з індекса k }
Var  i : word;
Begin
  SearchePos:=0;
  For i:=k to n do
    If x[i]>0 then
      Begin
        SearchePos:=i; Exit
      End;
End { SearchePos };
   { ------------------------------------------------ }
Begin
        У в е д е н н я    n, X
  i:=1;
  CondPos:=true;
  While CondPos do  { Перестановка додатних елементів }
    If x[i]>0 then  { на початок масиву }
      Begin
        Inc(i);
        If i>n then
          CondPos:=false;
      End
    Else
      Begin
        j:=SearchePos(i+1);
        If j=0 then
          CondPos:=false
        Else
          Begin
            R:=x[j];
            For k:=j downto i+1 do
              x[k]:=x[k-1];
```

```
            x[i]:=R;
          End;
      End;
    if i<=n then
      Begin
        CountZero:=0;
        For i:=n downto i do { підрахунок кількості }
          If x[i]=0 then     { нульових елементів і }
            Begin            { вилучення їх з масиву }
              Inc(CountZero);
              For j:=i to n-1 do
                x[j]:=x[j+1];
            End;
        If (CountZero>0) and (CountZero<n) then
          Begin
            For j:=n downto i+CountZero do  {Зрушення }
              x[j]:=x[j-CountZero];          { підмасиву}
            For j:=i to i+CountZero-1 do    { Запис нулів}
              x[j]:=0;
          End;
      End;
    В и в і д   X
End.
```

У програмах Cont3_1a і Cont3_1b для скорочення їхнього тексту інструментальні процедури не наведені.

Етап 2. Перетворення програми Cont3_1b у підпрограму в припущенні, що всі оброблювані масиви мають однакове ім'я типу, що в достатній мірі проілюстровано в програмі Cont3_2, яка приводиться нижче. Тут варто звернути увагу на наступне.

1. Уведення кожного з оброблюваних масивів *X,Y,Z* виконується з окремого файлу.

2. Уведення й друк оброблюваних масивів реалізується в основній програмі при звертанні до процедур `ReadArray, ScreenArray` і `PrinterArray.`

3. Всі інструментальні підпрограми незалежні від процедури Transpos і є внутрішніми тільки стосовно основної програми.

4. При обробці кожного з масивів *X, Y, Z* по описаному вище алгоритму відмінність полягає лише в імені й розмірі масиву; у зв'язку із цим для процедури Transpos в якості формальних параметрів взяте ім'я оброблюваного масиву *A* і його розмір *n*.

5. Опис змінних, наведений в розділі **Var** програми Cont3_1b, розділений на глобальні (розділ **Var** програми Cont3_2) і локальні (розділ **Var** процедури Transpos) змінні. До локальних змінних віднесені ті, які використовуються тільки в процедурі перестановки елементів масиву, що обробляється.

6. Процедура `ReadArray` повертає в якості вихідних параметрів ім'я масиву *A* и його розмір *n*. Тому перед іменами цих формальних параметрів записане слово **Var.**

7. При роботі процедури `ScreenArray` елементи формального масиву *A i* його розмір *n* не змінюються. Тому обоє ці параметри можна було б оголосити як параметри-значення. Однак у цьому випадку при звертанні до процедури була б зроблена передача всіх елементів фактичного масиву, у чому немає необхідності. Наявність слова **Var** тільки перед ім'ям формального масиву *A* приводить до того, що в

процедуру ScreenArray передаються не елементи масиву, а лише його адреса, що має розмір 4 байти.

```
Program Cont3_2;
Uses Crt,Printer;
Label  10;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  nx,ny,nz : word;          { розміри масивів X, Y, Z }
      IndPrinter : boolean;    { індикатор використання }
                               { принтера }
      ch : char;               { символ натиснутої клавіші }
      X,Y,Z : Ar;              { вхідні масиви }
      Fx,Fy,Fz : text;         { вхідні файли }
   { ---------------------------------------------- }
Procedure WaitEnter;
{ Затримка виконання програми, поки не буде }
{ натиснута клавіша Enter }
Var  ch : char;
Begin
  Writeln('Натисніть клавішу Enter');
  Repeat
    ch:=ReadKey;
  Until ord(ch)=13;
End { WaitEnter };
   { ---------------------------------------------- }
Procedure ReadArray(Var F:text; Var A:Ar; Var n:word);
{ Читання одновимірного масиву з текстового файлу }
Begin
  Reset(F);
  n:=0;
  While not SeekEof(F) do
    Begin
      Inc(n);
      Read(F,a[n]);
    End;
  Close(F);
End { ReadArray };
   { ---------------------------------------------- }
Procedure ScreenArray(S:string; Var A:Ar; n:word);
{ Вивід на екран одновимірного масиву }
Var  i,k : word;
Begin
  Writeln(S,'  n = ',n);
  k:=0;
   For i:=1 to n do
    Begin
      Inc(k);
      If k<10 then
        Write(a[i]:5,'  ')
      Else
        Begin
          k:=0;
```

```
          Writeln(a[i]:5);
        End;
    End;
  If k>0 then Writeln;
End { ScreenArray };
{ ---------------------------------------------- }
Procedure PrinterArray(S:string; Var A:Ar; n:word);
{ Вивід на принтер одновимірного масиву }
Var  i,k : word;
Begin
  Writeln(Lst,S,'  n = ',n);
  k:=0;
  For i:=1 to n do
    Begin
      Inc(k);
      If k<10 then
        Write(Lst,a[i]:5,'  ')
      Else
        Begin
          k:=0;
          Writeln(Lst,a[i]:5);
        End;
    End;
  If k>0 then Writeln(Lst);
End { PrinterArray };
   { ---------------------------------------------- }
Function SearchePos(Var A:Ar; n,k:word):word;
{ Пошук найближчого додатного елемента, }
{ починаючи з індексу k }
Var  i : word;
Begin
  SearchePos:=0;
  For i:=k to n do
    If a[i]>0 then
      Begin
        SearchePos:=i; Exit
      End;
End { SearchePos };
   { ---------------------------------------------- }
Procedure Transpos(Var A:Ar; n:word);
{ Перестановка елементів одновимірного масиву }
Var  i,j,k,               { параметри циклів }
    CountZero : word;     { лічильник нульових елементів }
    R : integer;          { буферна змінна }
    CondPos : boolean;    { змінна керування циклом }
Begin
  i:=1;
  CondPos:=true;
  While CondPos do        { Перестановка додатних }
    If a[i]>0 then        { елементів на початок масиву }
      Begin
        Inc(i);
        If i>n then
```

```
          CondPos:=false;
        End
      Else
        Begin
          j:=SearchePos(A,n,i+1);
          If j=0 then
            CondPos:=false
          Else
            Begin
              R:=a[j];
              For k:=j downto i+1 do
                a[k]:=a[k-1];
              a[i]:=R;
            End;
        End;
    If i<=n then
      Begin
        CountZero:=0;
        For i:=n downto i do     { Підрахунок кількості }
          If a[i]=0 then         { нульових елементів і }
            Begin                { вилучення їх з  }
              Inc(CountZero);    { масиву }
              For j:=i to n-1 do
                a[j]:=a[j+1];
            End;
        If CountZero<n then
          Begin
            For j:=n downto i+CountZero do { Зрушення підмасиву }
              a[j]:=a[j-CountZero];
            For j:=i to i+CountZero-1 do   { Запис нулів}
              a[j]:=0;
          End;
      End;
End { Transpos };
{ ------------------------------------------------ }
Begin

{ Установлення відповідності між внутрішніми }
{ і зовнішніми файлами }
  Assign(Fx,'E:\kontr2\x.dat');
  Assign(Fy,'E:\kontr2\y.dat');
  Assign(Fz,'E:\kontr2\z.dat');

{ Запит про використання принтера }
  ClrScr;
  Writeln(' Чи буде використаний принтер (Так,Ні) ?');
  ch:=ReadKey;
  If  ch in ['Д','д','L','l'] then
    IndPrinter:=true
  Else
    IndPrinter:=false;
  ClrScr;
```

```
{ Обробка масиву X }
  ReadArray(Fx,X,nx);
  ScreenArray('Вхідний масив X',X,nx);
  If  IndPrinter then
    PrinterArray('Вхідний масив X',X,nx);
  Transpos(X,nx);
  ScreenArray('Перетворений масив X',X,nx);
  If IndPrinter then
    PrinterArray('Перетворений масив X',X,nx);

{ Обробка масиву Y }
  ReadArray(Fy,Y,ny);
  ScreenArray(' Вхідний масив Y',Y,ny);
  If  IndPrinter then
    PrinterArray(' Вхідний масив Y',Y,ny);
  Transpos(Y,ny);
  ScreenArray('Перетворений масив Y',Y,ny);
  If IndPrinter then
    PrinterArray('Перетворений масив Y',Y,ny);

{ Обробка масиву Z }
  ReadArray(Fz,Z,nz);
  ScreenArray(' Вхідний масив Z',Z,nz);
  If  IndPrinter then
    PrinterArray(' Вхідний масив Z',Z,nz);
  Transpos(Z,nz);
  ScreenArray('Перетворений масив Z',Z,nz);
  If  IndPrinter then
    PrinterArray('Перетворений масив Z',Z,nz);
  WaitEnter;
End.
```

Етап 3. На даному етапі виконується перетворення програми Cont_3b у припущенні, що оброблювані масиви `X,Y,Z` мають різні імена типів.

Практично для кожної мови програмування створюються пакети прикладних програм (ППП), які реалізують різноманітні процедури чисельного аналізу (рішення нелінійних рівнянь, визначення коренів поліномів, операції над матрицями, рішення системи диференціальних рівнянь і т.п.). Прикладна програма розробляється у вигляді підпрограми, звертання до якої виконується із програми користувача. При цьому, як правило, ППП поставляється у вигляді об'єктних модулів, що виключає можливість зміни їхнього тексту користувачем.

Паскаль-підпрограма, що обробляє масив, повинна в списку формальних параметрів містити ім'я цього масиву із вказівкою відповідного імені типу. У загальному випадку ім'я типу формального масиву не збігається з ім'ям типу фактичного масиву в програмі користувача. Таким чином, прикладна підпрограма повинна забезпечувати сумісність типів формального й фактичного масивів.

У Паскаль-програмі масив завжди має фіксований розмір, обумовлений його ім'ям типу. Необхідність використання різних імен типів зв'язана головним чином з тим, що формальний і фактичний масиви в загальному випадку мають різні розміри. Типи елементів цих масивів, природно, повинні бути однаковими.

Припустимо, що для масивів *X,Y* і *Z*, які мають різні імена типів, необхідно обчислити середнє арифметичне їхніх елементів. Тоді програма може мати такий вигляд:

```
Program Example1;
Type  Xar = array[1..50] of real;
      Yar = array[1..500] of real;
      Zar = array[1..5000] of real;
Var i,nx,ny,nz : word;
     Sx,Sy,Sz : real;
     X : Xar;
     Y : Yar;
     Z : Zar;
Procedure MiddleAr(Var Buf:Xar; Var S:real; n:word);
Var  i : word;
Begin
  S:=0;
  For i:=1 to n do
    S:=S+Buf[i];
  S:=S/n;
End { MiddleAr };
Begin
  Уведення і друк nx,ny,nz,X,Y,Z
  MiddleAr(X,Sx,nx);
  MiddleAr(Y,Sy,ny);
  MiddleAr(Z,Sz,nz);
  Друк Sx,Sy,Sz
End.
```

Тут при трансляції програми буде правильно сприйняте лише перше звертання до процедури MiddleAr; для інших звертань буде видане повідомлення про невідповідність типів фактичних масивів *Y, Z* і формального масиву *Buf.*

Можливість обробки масивів з різними іменами типів може бути забезпечена використанням абсолютних змінних.

У Турбо Паскалі допускаються формальні параметри без типу. Таким формальним параметрам можуть відповідати фактичні параметри будь-якого типу. Це дозволяє за допомогою апарата абсолютних змінних організувати в процедурі обробку масивів різного розміру. Зокрема, програма Example у цьому випадку буде мати такий вигляд:

```
Program Example2;
Type  Xar = array[1..50] of real;
     Yar = array[1..500] of real;
     Zar = array[1..5000] of real;
Var i,nx,ny,nz : word;
     Sx,Sy,Sz : real;
     X : Xar;
     Y : Yar;
     Z : Zar;
Procedure MiddleAr(Var Buf; Var S:real; n:word);
Type  RealAr = array[1..10000] of real;
Var  i : word;
     A : RealAr absolute Buf;
Begin
  S:=0;
  For i:=1 to n do
```

```
    S:=S+a[i];
   S:=S/n;
 End { MiddleAr };
 Begin
   Уведення і друк nx,ny,nz,X,Y,Z
   MiddleAr(X,Sx,nx);
   MiddleAr(Y,Sy,ny);
   MiddleAr(Z,Sz,nz);
   Друк  Sx,Sy,Sz
 End.
```

У розділі **Var** процедури MiddleAr оголошена змінна A типу RealAr. Проте цієї змінної, на відміну від інших локальних змінних, пам'ять не виділяється. Їй привласнюється та ж адреса, що одержує формальна змінна Buf при звертанні до процедури MiddleAr. Отже, при першому звертанні до цієї процедури імена A, Buf, X - це три імені того самого поля пам'яті, а оскільки тип змінної A зазначений явно, то програма має всю необхідну інформацію для обробки поля X, безпосередньо використовуючи для цього ім'я A.

У програмі Cont3_3, що реалізує обробку масивів з різними іменами типів, у процедурах ReadArray, ScreenArray та інших на формальний параметр A без імені типу накладається масив з типом RealAr. При звертанні до процедури формальне ім'я A замінюється на ім'я фактичного масиву. Оскільки RealAr - це ім'я типу, а не опис змінної, то об'єкту з ім'ям RealAr ніякої пам'яті не виділяється. Як і будь-яке ім'я типу, RealAr визначає безліч значень, які має право приймати змінна із цим ім'ям типу. Оскільки масив не може займати більш ніж 64 Кбайт (65536 байт) пам'яті, то для типу елемента real максимальний розмір масиву може мати опис

```
RealAr = array[1..10922] of real або
RealAr = array[1..2*MaxInt div SizeOf(real)] of real.
```

```
Program Cont3_3;
{ Етап 3 з використанням абсолютних змінних }
Uses Crt,Printer;
Label  10;
Const  NmaxX = 500; NmaxY = 100; NmaxZ = 300;
Type ArX = array[1..NmaxX] of integer;
     ArY = array[1..NmaxY] of integer;
     ArZ = array[1..NmaxZ] of integer;
     IntAr = array[1..2*MaxInt div SizeOf(integer)] of
                                          integer;
Var   nx,ny,nz : word;     { розмір масивів X, Y, Z }
    IndPrinter : boolean; { індикатор використання }
                          { принтера }
    ch : char;            { символ натиснутої клавіші }
    X : ArX;              { вхідний масив X }
    Y : ArY;              { вхідний масив Y }
    Z : ArZ;              { вхідний масив Z }
    Fx,Fy,Fz : text;      { вхідні файли }
{ ------------------------------------------------ }
Procedure WaitEnter;
{ Затримка виконання програми доти, поки не буде }
{ натиснута клавіша Enter }
```

```
Var  ch : char;
Begin
  Writeln('Натисніть клавішу Enter');
  Repeat
    ch:=ReadKey;
  Until ord(ch)=13;
End { WaitEnter };
{ ------------------------------------------------ }
Procedure ReadArray(Var F:text; Var A; Var n:word);
{ Читання одномірного масиву з текстового файлу }
Var  B : IntAr absolute A;
Begin
  Reset(F);
  n:=0;
  While not SeekEof(F) do
    Begin
      Inc(n);
      Read(F,b[n]);
    End;
  Close(F);
End { ReadArray };
{ ------------------------------------------------ }
Procedure ScreenArray(S:string; Var A; n:word);
{ Вивід на екран одномірного масиву }
Var  i,k : word;
    B : IntAr absolute A;
Begin
  Writeln(S,'  n = ',n);
  k:=0;
  For i:=1 to n do
    Begin
      Inc(k);
      If k<10 then
        Write(b[i]:5,'   ')
      Else
        Begin
          k:=0;
          Writeln(b[i]:5);
        End;
    End;
  If k>0 then Writeln;
End { ScreenArray };
{ ------------------------------------------------ }
Procedure PrinterArray(S:string; Var A; n:word);
{ Вивід на принтер одномірного масиву }
Var  i,k : word;
     B : IntAr absolute A;
Begin
  Writeln(Lst,S,'  n = ',n);
  k:=0;
  For i:=1 to n do
    Begin
      Inc(k);
```

```
      If k<10 then
        Write(Lst,b[i]:5,'   ')
      Else
        Begin
          k:=0;
          Writeln(Lst,b[i]:5);
        End;
    End;
  If k>0 then Writeln(Lst);
End { PrinterArray };
{ ---------------------------------------------- }
Function SearchePos(Var A; n,k:word):word;
{ Пошук найближчого позитивного елемента, починаючи }
{ з індексу k }
Var  i : word;
     B : IntAr absolute A;
Begin
  SearchePos:=0;
  For i:=k to n do
    If  b[i]>0 then
      Begin
        SearchePos:=i; Exit
      End;
End { SearchePos };
{ ---------------------------------------------- }
Procedure Transpos(Var A; n:word);
{ Перестановка елементів одномірного масиву }
Var   i,j,k,           { параметри циклів }
CountZero : word;      { лічильник нульових елементів }
R : integer;           { буферна змінна }
CondPos : boolean;     { змінна керування циклом }
B : IntAr absolute A;
Begin
  i:=1;
  CondPos:=true;
  While CondPos do       { Перестановка позитивних }
    If  b[i]>0 then      { елементів на початок масиву }
      Begin
        Inc(i);
        If i>n then
          CondPos:=false;
      End
    Else
      Begin
        j:=SearchePos(A,n,i+1);
        If j=0 then
          CondPos:=false
        Else
          Begin
            R:=b[j];
            For k:=j downto i+1 do
              b[k]:=b[k-1];
            b[i]:=R;
```

```
          End;
        End;
      If  i<=n then
      Begin
        CountZero:=0;
        For i:=n downto i do  { Підрахунок кількості }
          If  b[i]=0 then     { нульових елементів й }
            Begin             { видалення їх з масиву }
              Inc(CountZero);
              For j:=i to n-1 do
                b[j]:=b[j+1];
            End;
        If CountZero<n then
          Begin
            For j:=n downto i+CountZero do { Зрушення подмассива }
              b[j]:=b[j-CountZero];
            For j:=i to i+CountZero-1 do   { Запис нулів }
              b[j]:=0;
          End;
      End;
End { Transpos };
{ ------------------------------------------------ }
Begin
{ Установлення відповідності між внутрішніми }
{ і зовнішніми файлами }
  Assign(Fx,'E:\Lab2\x.dat');
  Assign(Fy,'E:\Lab2\y.dat');
  Assign(Fz,'E:\Lab2\z.dat');

{ Запит про використання принтера }
  ClrScr;
  Writeln(' Чи буде використаний принтер (Так,Ні) ?');
  ch:=ReadKey;
  If  ch in ['Д','д','L','l'] then
    IndPrinter:=true
  Else
    IndPrinter:=false;

{ Обробка масиву X }
  ClrScr;
  ReadArray(Fx,X,nx);
  ScreenArray('Вхідний масив X',X,nx);
  If  IndPrinter then
    PrinterArray('B

хідний масив X',X,nx);
  Transpos(X,nx);
  ScreenArray('Перетворений масив X',X,nx);
  If  IndPrinter then
    PrinterArray('Перетворений масив X',X,nx);
  WaitEnter;

{ Обробка масиву Y }
```

```
  ClrScr;
  ReadArray(Fy,Y,ny);
  ScreenArray('Вхідний масив Y',Y,ny);
  If  IndPrinter then
    PrinterArray('Вхідний масив Y',Y,ny);
  Transpos(Y,ny);
  ScreenArray('Перетворений масив Y',Y,ny);
  If  IndPrinter then
    PrinterArray('Перетворений масив Y',Y,ny);
  WaitEnter;

{ Обробка масиву Z }
  ClrScr;
  ReadArray(Fz,Z,nz);
  ScreenArray('Вхідний масив Z',Z,nz);
  If  IndPrinter then
    PrinterArray('Вхідний масив Z',Z,nz);
  Transpos(Z,nz);
  ScreenArray('Перетворений масив Z',Z,nz);
  If  IndPrinter then
    PrinterArray('Перетворений масив Z',Z,nz);
  WaitEnter;

End.
```

Результати роботи програми:

```
                    Вхідний масив X   n=22
10     20      6     30     60     40     50     100    80     90
70     50     34     44     54     64     75     65     55     45
35     25
                    Перетворений масив X    n=22
10     20      6     30     60     40     50     100    80     90
70     50     34     44     54     64     75     65     55     45
35     25
                    Вхідний масив Y    n=24
10     20     -6     30    -60    -40     50    -100    80     90
70    -50     34     44     54    -64    -74     -75    77     48
-39   -25     25     23
                    Перетворений масив Y    n=24
 10     20     30     50     80     90     70      34    44     54
 77     48     25     23     -6    -60    -40    -100   -50    -64
-74    -75    -39    -25
                    Вхідний масив Z    n=33
10      20     -6      0      0      0     30    -60    -40     0
50    -100      0     80     90     70    -50     34     44     0
 0      54    -64    -74    -75     77     48    -39    -25    25
23       0      0
                    Перетворений масив Z    n=33
 10     20     30     50     80     90     70     34     44     54
 77     48     25     23      0      0      0      0      0      0
  0      0      0     -6    -60    -40    -100   -50    -64    -74
-75    -39    -25
```

У завершення розглянемо питання про використання буферних масивів у процедурах, які призначені для обробки масивів з різними іменами типів.

Цілком очевидно, що програма Cont3_1a більш проста в порівнянні із програмою Cont3_1b. Разом з тим програма Cont3_1a більш ефективна по швидкодії, оскільки в ній не відбуваються численні зрушення підмасивів. Єдиним недоліком цієї програми є використання в ній додаткових (буферних) масивів Xpos і Xneg. Цей недолік, цілком прийнятний при обробці одного масиву, є серйозною перешкодою при перетворенні позначеної програми в процедуру, призначену для обробки масивів з різними іменами типів.

Кількість елементів у кожному з масивів Xpos і Xneg повинне бути таким же, як й у вхідному масиві X (в окремому випадку масив X може складатися тільки з додатних або тільки з від'ємних елементів, тоді повністю буде заповнений тільки один з буферних масивів, а другий залишиться порожнім).

Якщо в процедурі, аналогічній Transpos, але такої, що реалізує алгоритм програми Cont3_2, оголосити масиви Xpos і Xneg локальними, то стає неясним питання про тип цих масивів. Формальний масив A у процедурі Transpos не вимагає для себе окремого поля пам'яті, він сполучається за адресою з оброблюваним масивом X, Y або Z. Для буферних масивів потрібні окремі поля пам'яті. У цьому випадку довелося б указати для них тип максимальний їз оброблюваних масивів, що не сприяло б, зокрема, універсальності розробленої процедури. Перенесення оголошення буферних масивів у глобальний розділ опису змінних не поліпшило б ситуацію з неефективним використанням пам'яті. У цьому випадку довелося б оголосити або три пари буферних масивів відповідно з типами ArX, ArY і ArZ, або, як і при локальному описі, оголосити одну пару з максимальним типом Ar.

У зв'язку із труднощами, які виникають при використанні буферних масивів у процедурах обробки масивів з різними іменами типів, у загальних зауваженнях до варіантів завдання підкреслюється, що при їх програмній реалізації буферні масиви не повинні застосовуватися.

В а р і а н т и з а в д а н ь

1. Задано масив цілих чисел $X = (x_1, x_2, ..., x_n)$. Сформувати масив $Y = (y_1, y_2, ..., y_m)$, помістивши в нього в порядку зменшення всі різні числа, які містяться в масиві *X*. Визначити, наскільки відрізняються середні арифметичні значення елементів масивів *X* й *Y*.

Примітка. "Різні" й "неповторювальні" числа – це не те саме. Наприклад, у масиві *X* = (5, 8, 18, -3, 10, 14, 14, 8, 12, 7, 5, 12, 8, 4, 12) різні числа – це масив *Y* = (18, 14, 12, 10, 8, 7, 5, 4, -3), а неповторювальні – масив *Z* = (18, 10, 7, 4, -3).

Рекомендація.

1) Записати $y_1 = x_1$; $m = 1$.

2) Переглядаючи елементи x_i (i = 2 .. n), визначити, чи є елемент x_i у масиві *Y*. Якщо такий елемент не виявлений, то додати його в масив *Y*, збільшуючи при цьому значення змінної *m*.

---

2. Масив $X = (x_1, x_2, ..., x_n)$ містить велику кількість нульових елементів. Визначити положення й розмір найбільш довгої серії таких елементів і вилучити її із складу масиву.

---

3. Задано два масиви $X = (x_1, x_2, ..., x_n)$ й $Y = (y_1, y_2, ..., y_m)$, до складу яких входять натуральні числа, причому в кожному із цих масивів немає повторюваних елементів.

Сформувати масив *Z*, включивши в нього всі елементи, які одночасно втримуються в масивах *X* й *Y*.

Рекомендація.

Перебираючи елементи масиву *Y*, визначити, чи є елемент y_i у масиві *X*, і, якщо такий елемент виявлений, додати його в масив *Z*.

---

4. Задано цілочисельний масив $X=(x_1,x_2,...,x_n)$, у якому можуть бути однакові числа. Знайти максимальний і мінімальний елементи серед неповторюваних чисел і обміняти їх місцями. Урахувати окремі випадки:

- у масиві немає неповторюваних чисел;
- масив містить лише одне неповторюване число.

---

5. З масиву цілих додатних чисел $X=(x_1,x_2,...,x_n)$ вилучити всі парні за значенням елементи, крім останнього. Числа, які залишилися після вилучення парних елементів, розташувати в порядку зростання. Урахувати окремі випадки (у масиві немає парних елементів, є тільки один парний елемент, всі елементи - парні). Буферний масив не використовувати.

Примітка. Масив *X* рекомендується переглядати справа наліво. Тоді перший зустрінутий парний елемент не буде підлягати вилученню, інші парні елементи повинні бути вилучені.

---

6. Задано дійсні масиви $X=(x_1,x_2,...,x_n)$ і $Y=(y_1,y_2,...,y_n)$. Вважаючи елементи (x_i,y_i) координатами точок на площині, визначити, чи є в масивах *X, Y* три суміжні точки (1,2,3), (2,3,4), (3,4,5),..., які лежать на одній прямій. Надрукувати номер першої і останньої груп таких точок (якщо вони є). Із складу першої і останньої груп вилучити першу точку.

Примітка. Використати наступну властивість: якщо три точки лежать на одній прямій, то площа трикутника з вершинами в цих точках дорівнює нулю. Подвоєна площа трикутника

$$S = x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)\ .$$

Площу трикутника вважати нульовою, якщо $|S| < \varepsilon$, де ε - мале число (наприклад, 0.001).

Примітка. В умові завдання передбачається, що на одній прямій може бути розташовано три і більше точок.

---

7. У масиві $X=(x_1,x_2,...,x_n)$ переставити місцями перший і другий від'ємні елементи, третій і четвертий від'ємні елементи і т.д. Якщо кількість від'ємних елементів у масиві менше двох, перетворення масиву не виконувати. Визначити, як змінилося положення мінімального й максимального елементів масиву *X* при його перетворенні.

Рекомендація.

Тут доцільно створити процедуру Searche (`Var k,k1,k2:integer`), де *k* - індекс початку пошуку від'ємного елемента, *k*1 і *k*2 – індекси двох найближчих від'ємних елементів. У початковій частині процедури `Searche` установити $k1 = 0$ і $k2 = 0$. Обмін елементів масиву *X* виконується, якщо $k1 \neq 0$ і $k2 \neq 0$. Якщо ж $k1 = 0$ або $k2 = 0$, то подальший аналіз масиву *X* припиняється.

---

8. Задано дійсний масив $X=(x_1,x_2,...,x_n)$, його елементи – це значення кутів у радіанах. Сформувати для кожного значення x_i елементи цілочисельних масивів g_i, m_i, s_i, які визначають ці ж кути в градусах, мінутах і секундах.

Примітка. Секунди повинні бути округлені до найближчого цілого значення. Інтервали значень: $0 \le g_i < 360$; $0 \le m_i < 60$; $0 \le s_i < 60$. Отже, якщо вхідний кут $x_i > 2\pi$, те його попередньо потрібно привести до значення $x_i \le 2\pi$.

Рекомендація.

Позначимо через $d = x_i$ вхідне значення кута в радіанах, через b – те ж у градусах (d *i b* – дійсні значення), через g, m, s – цілочисельні значення градусів, мінут і секунд. Тоді необхідне перетворення можна виконати в наступному порядку:

```
b:=180*d/pi
g:=trunc(b)
b:=180*(b-g)/pi
m:=trunc(b)
b:=180*(b-m)/pi
s:=round(b)
```

---

9. Визначити найбільший загальний дільник всіх чисел, які містяться в заданій послідовності $X=(x_1,x_2,...,x_n)$ цілих додатних чисел.

Примітка.

Найбільший загальний дільник $d(a,b)$ для масиву $X=(x_1,x_2,...,x_n)$ визначається по алгоритму Евкліда. Послідовність дій:

$$p = d(x_1, x_2)$$
$$p = d(p, x_3)$$
$$\cdots\cdots\cdots\cdots\cdots$$
$$p = d(p, x_n)$$

---

10. У цілочисельному масиві $X=(x_1,x_2,...,x_n)$ вилучити всі непарні елементи, крім останнього такого елемента. Визначити, як при цьому змінилися середнє арифметичне значення S і середнє квадратичне відхилення G елементів масиву X (у відсотках). Буферний масив не використовувати.

---

11. Елементи масиву $X=(x_1,x_2,...,x_n)$ - це послідовність цифр цілого числа, записаного в системі числення з основою q, $2 \le q \le 10$, $0 \le x_i < q$. Переставити цифри числа у зворотному порядку, вилучити незначущі нулі (якщо вони є) і надрукувати десяткове значення цього числа до і після перестановки. Значення q увести із клавіатури.

Вказівка. Десяткове значення числа обчислювати за схемою Горнера.

Рекомендація.

Припустимо, що після перестановки цифр числа й вилучення незначущих нулів ми одержали

$$X = (x_1, x_2, ..., x_m),$$

де x_i – цифри числа в системі числення з основою q. Тоді це число можна представити у вигляді

$$P = x_1 q^{m-1} + x_2 q^{m-2} + x_3 q^{m-3} + ... + x_{m-1} q + x_m$$

або

$$P = (...(x_1 q + x_2) q + x_3) q + ... + x_{m-1}) q + x_m$$

Останній вираз - це і є схема Горнера для обчислення полінома.

12. Відомо, що в цілочисельному масиві $X=(x_1,x_2,...,x_n)$ три й тільки три числа рівні між собою. Знайти ці числа й перемістити їх у початок масиву, зрушивши інші числа до кінця цього масиву.

---

13. За однократний перегляд цілочисельного масиву $X=(x_1,x_2,...,x_n)$ знайти його максимальний додатний елемент $x_{\max}$ і визначити середнє арифметичне значення всіх елементів масиву, за винятком елементів, рівних $x_{\max}$ (у загальному випадку в масиві може бути кілька елементів, рівних $x_{\max}$).

Вказівка. У програмі повинні бути враховані окремі випадки, у тому числі
- у масиві немає додатних елементів;
- всі елементи масиву додатні й рівні один одному.

---

14. У цілочисельному масиві $X=(x_1,x_2,...,x_n)$ кожну пару x_i і x_j ($i,j=1,2,...,n;\ i\neq j$) непарних елементів перетворити в парні елементи по формулам: $x_i := x_i+1,\ x_j := x_j-1$. Пари елементів i, j вибирати в порядку їхнього проходження в масиві *X*. Визначити, як при цьому змінилися максимальний і мінімальний елементи масиву *X* (в абсолютному відношенні і в відсотках).

Рекомендація.

Розробити процедуру `Transform(`**`Var`** `k,k1,k2:integer),` де k — початок пошуку непарних елементів, *k*1 і *k*2 – індекси найближчих непарних елементів. У початковій частині процедури встановити *k*1 = 0 і *k*2 = 0. При першому звертанні до процедури `Transform` задати *k* = 1, при наступних обігах *k* = *k*2+1. Якщо після закінчення роботи процедури `Transform` буде отримано *k*1 = 0 або *k*2 = 0, то подальший аналіз масиву *X* припинити.

---

15. Елементи кожної серії додатних чисел масиву $X=(x_1,x_2,...,x_n)$ згрупувати в порядку зменшення. До складу серії повинно входити не менше трьох елементів.

---

16. По заданому цілочисельному масиві $X=(x_1,x_2,...,x_n)$ сформувати масив $Y=(y_1,y_2,...,y_n)$ такий, що y_i - це кількість елементів масиву *X*, які не перевищують значення x_i (тобто менше або дорівнює значенню x_i) на кінцевому відрізку цього масиву від елемента з індексом *i*+1 до елемента з індексом *n*. Цілком очевидно, що для y_n повинно бути записане нульове значення.

Приклад. Для масиву

X = (5, 8, 3, 7, 5, 4, 10, 8, 12, 15, 3, 7)

будемо мати

Y = (4, 7, 1, 4, 2, 1, 3, 2, 2, 2, 0, 0).

---

17. Задано два цілочисельних масиви $X=(x_1,x_2,...,x_n)$ і $Y=(y_1,y_2,...,y_m)$. Нехай у масиві *X* міститься k_1 парних елементів, а в масиві *Y* - k_2 непарних елементів. Обміняти місцями $k=\min(k_1,k_2)$ парних елементів масиву *X* с непарними елементами масиву *Y* (у порядку їхнього проходження в масивах *X* и *Y*). Врахувати, що в окремому випадку може бути *k* = 0.

---

18. Перетворити масив $X=(x_1,x_2,...,x_n)$, розташувавши спочатку його від'ємні, а потім невід'ємні елементи, зберігши при цьому в групі від'ємних елементів їх первісний відносний порядок, а в групі невід'ємних елементів змінивши його на зворотний. Визначити, як при цьому змінилося положення мінімального по модулю елемента масиву *X*. Буферний масив не використовувати.

Приклад.

Для масиву

$$X = (5, 8, -3, 0, -4, 7, -10, -6, 0, -1)$$

одержимо

$$X = (-3, -4, -10, -6, -1, 5, 8, 0, 7, 0).$$

---

19. Заданий дійсний масив $X=(x_1,x_2,...,x_n)$ осереднити у такий спосіб: максимальний і мінімальний елементи замінити їх середнім арифметичним значенням, то ж зробити щодо максимального й мінімального елементів перетвореного масиву *X* і т.д. Якщо в черговому циклі обробки масиву *X* виявиться, що його максимальний і мінімальний елементи відрізняються між собою не більше ніж на значення ε (ε - досить мала величина), то подальше перетворення масиву *X* не виконувати. Визначити, як змінилися середнє арифметичне значення *S* і середнє квадратичне відхилення *G* елементів масиву *X* після його перетворення. Зміни параметрів *S* і *G* виразити у відсотках, з урахуванням знака відхилення.

$$S=\frac{1}{n}\sum_{i=1}^{n}x_i\ ;\quad G=\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-S)^2}$$

---

20. Елементи масиву $Y=(y_1,y_2,...,y_m)$ - це ординати точок ламаної лінії в рівновіддалених вузлах по осі *X*. Серед "зубців" ламаної, для яких виконується відношення $y_{i-1}<y_i>y_{i+1}$, знайти "вершину зубця" з максимальним перевищенням над сусідніми точками $\Delta y=y_i-\frac{y_{i-1}+y_{i+1}}{2}$ й вилучити її зі складу ламаної. Визначити, як при цьому змінилася загальна довжина ламаної лінії (вважати, що відстань між вузлами по осі *X* дорівнює 1). Зміна довжини ламаної виразити у відсотках.

Примітка.

Довжина відрізка ламаної при кроці $h=1$ по осі абсцис

$$l_i=\sqrt{(y_{i+1}-y_i)^2+1}$$

---

21. Визначити, чи містяться в заданій послідовності цілих додатних чисел $X=(x_1,x_2,...,x_n)$ числа Фібоначчі і, якщо це так, визначити значення й положення максимального й мінімального з таких чисел, після чого обміняти їх місцями.

Вказівка. У програмі використати процедуру генерації чисел Фібоначчі, що не перевищують значення аналізованого елемента x_i.

Числа Фібоначчі одержують по формулі

$$f_{i+2}=f_{i+1}+f_i;\qquad f_1=1;\quad f_2=1$$

---

22. Вилучити з дійсного масиву $X=(x_1,x_2,...,x_n)$ всі елементи, які перевищують його середнє арифметичне значення *S*, крім першого такого елемента, і визначити, як при цьому змінилося значення *S* (у відсотках). Буферний масив не використовувати.

---

23. При однократному перегляді цілочисельного масиву $X=(x_1,x_2,...,x_n)$ знайти два максимальних за значенням елемента, кратних одночасно числам 2 і 3, і, якщо такі елементи існують, обміняти їх місцями в масиві.

Примітка. Якщо максимальний елемент кратний одночасно числам 2 і 3, необхідно зробити пошук іншого елемента, кратного числам 2 і 3. Обмін не робити, якщо

- у масиві менше двох елементів, кратних числам 2 і 3;
- два максимальних елементи, які кратні одночасно числам 2 і 3, у той же час рівні один одному..

Вказівка.

Для перевірки подільності числа на 2 використати функцію `odd`, для перевірки подільності на 3 – операцію **`mod`**.

---

24. У масиві $X=(x_1,x_2,...,x_n)$ визначити положення й розмір першої і останньої серії додатних елементів, до складу яких входить не менш чотирьох елементів, і якщо ці серії існують, то вилучити їх із складу масиву.

---

25. Всі додатні числа в дійсному масиві $X=(x_1,x_2,...,x_n)$ переставити у зворотному порядку, не змінюючи положення інших чисел. Буферний масив не використовувати.

Примітка. Якщо в аналізованій парі x_i і x_j має місце $x_i=x_j$, то очевидно, що обмін таких елементів не має сенсу.

Рекомендація.

Тут доцільно розробити дві функції: `FromLeft(k1:integer): integer` і `FromRight(k2:integer):integer`. Перша з них виконує пошук найближчого справа додатного числа, починаючи з позиції *k*1; друга – найближчого зліва, починаючи з позиції *k*2. Пошук припиняється, коли $k1\geq k2$.

---

26. Дійсний масив $X=(x_1,x_2,...,x_n)$ містить кілька від'ємних елементів, які розділяють його на окремі підмасиви. Згрупувати елементи кожного підмасива в порядку зростання. Врахувати окремі випадки (у масиві немає від'ємних елементів; підмасив порожній або містить тільки один елемент).

Рекомендація.

Тут доцільно для групування підмасива окремо розробити процедуру `Group(k1,k2:integer)`, де *k*1 і *k*2 – початок і кінець підмасива, для якого виконується групування. Крім того, повинна бути функція `Negativ(k:integer):integer`, що визначає положення найближчого від'ємного числа, починаючи з позиції *k*.

---

27. Задано масив $X=(x_1,x_2,...,x_n)$, елементами якого є натуральні числа. У складі масиву *X* можуть бути повторювані елементи. Сформувати масив *Y*, включивши в нього всі неповторювані елементи з масиву *X*.

Рекомендація. Доцільно в програмі передбачити функцію `Repeat(k:integer):boolean`, де *k* – індекс елемента масиву, що перевіряється. Якщо вихідне значення цієї функції дорівнює `false`, то такий елемент потрібно включити до складу формованого масиву *Y*.

---

28. У дійсному масиві $X=(x_1,x_2,...,x_n)$ кожну пару x_i і x_j $(i,j=1,2,...,n;\ i\neq j)$ від'ємних елементів перетворити в додатні по формулі $x_i,x_j=\sqrt{x_i\,x_j}$. Пари елементів

(i, j) вибирати в порядку їхнього проходження в масиві *X*. Визначити, як при цьому змінилося середнє арифметичне значення елементів масиву *X* (у відсотках).

---

29. Значення цілої частини невід'ємного дійсного числа задано в десятковій системі числення у вигляді масиву цифр $A = (a_n, a_{n-1}, ..., a_1)$, значення його дробової частини - у вигляді масиву цифр $B = (b_1, b_2, ..., b_m)$. Виконати округлення числа, залишивши в ньому *k* дробових цифр ($0 \le k < m$), і надрукувати отримане десяткове значення. Параметр *k* увести із клавіатури.

Рекомендація.

Якщо в $b_{k+1} \ge 5$, то

1) відкинути розряди *k*+1, *k*+2, …, *m;*

2) число $a_n a_{n-1}..a_1.b_1 b_2..b_k$ скласти з одиницею в *k-ому* розряді.

Якщо $b_{k+1} < 5$, то додавання не робити.

---

30. Задано два масиви $X = (x_1, x_2, ..., x_n)$ і $Y = (y_1, y_2, ..., y_m)$, до складу яких входять натуральні числа, причому в кожному із цих масивів немає повторюваних елементів. Сформувати масив *Z*, об'єднавши масиви *X* й *Y*, при цьому в масиві *Z* також не повинно бути повторюваних елементів.

Рекомендація.

1) Переписати в масив *Z* всі елементи масиву *X*.

2) Переглядаючи елементи $y_i, i = 1..m$, визначити, чи є в масиві *X* елемент, що дорівнює значенню y_i. Якщо це не виявлено, то додати елемент y_i до складу масиву *Z*.

---

31. Задано дійсний масив $X = (x_1, x_2, ..., x_n)$, його елементи - це значення відстаней у милях. Потрібно сформувати для кожного значення x_i елементи цілочисельних масивів k_i, m_i, s_i, які визначають відстань у кілометрах, метрах і сантиметрах.

Примітка. 1 миля = 1,609344 км.

Сантиметри повинні бути округлені до найближчого цілого значення.

Див. рекомендації до п.8.

---

32. Елементи масиву $X = (x_1, x_2, ..., x_n)$ - це послідовність цифр цілого числа, записаного в системі числення з основою *q*, $2 \le q \le 10$, $0 \le x_i < q$. Згрупувати цифри числа в порядку зменшення й надрукувати десяткове значення цього числа до і після угруповання. Значення *q* увести із клавіатури.

Вказівка. Десяткове значення числа обчислювати за схемою Горнера.

Див.рекомендації до п.11.

---

33. У заданому масиві цілих додатних чисел $X = (x_1, x_2, ..., x_n)$ знайти найбільший підмасив, що є перестановкою деякого відрізка ряду натуральних чисел. До складу підмасива повинно входити не менше трьох елементів. Наприклад, таким відрізком є послідовність (18, 21, 16, 19, 14, 22, 15, 20, 17). Якщо заданий підмасив знайдений, то вилучити його із вхідного масиву.

Примітка.

Підмасив є перестановкою деякого відрізка натурального ряду чисел, якщо

1) кількість елементів підмасива

$$k = x_{\max} - x_{\min} + 1,$$

де $x_{\max}$, $x_{\min}$ – максимальний і мінімальний елементи підмасива;

2) у підмасиві немає однакових чисел.

---

34. У цілочисельному масиві $X=(x_1,x_2,...,x_n)$, що не містить однакових елементів, вилучити m найменших елементів, а елементи, що залишилися при цьому, згрупувати по зростанню. Значення m увести із клавіатури, $m<<n$.

Вказівка. Рекомендується спочатку згрупувати вхідний масив по зростанню, а потім вилучити з нього перші m елементів.

---

35. Для заданого значення n сформувати масив $X=(x_1,x_2,...,x_n)$ як послідовність чисел Фібоначчі. Оцінити, наскільки відрізняється положення елемента x_i, найбільш близького до середнього арифметичного значення S масиву X, від положення елемента x_j, найбільш близького до середнього геометричного значення P цього ж масиву.

Примітка.

$$S=\frac{1}{n}\sum_{i=1}^{n}x_i\ ;\qquad P=\sqrt[n]{\prod_{i=1}^{n}x_i}$$

---

36. З масиву цілих додатних чисел $X=(x_1,x_2,...,x_n)$ вилучити всі непарні елементи, крім першого такого елемента, після чого числа, які залишилися, розташувати в порядку зменшення. Буферний масив не використати.

---

37. Числова послідовність формується за правилом:

$u_0=\cos(x);\ u_1=\cos(x+h);\ u_2=\cos(x+2h);...;\ u_n=\cos(x+nh)$

(значення x, n, h задані). Серед тих елементів послідовності $U=(u_0,u_1,...,u_n)$, які перевищують по модулю задане значення b, знайти максимальний і мінімальний елементи, після чого обміняти їх місцями. Врахувати, що в окремому випадку таких елементів може бути менше двох. Значення x, n, h, b повинні бути уведені із клавіатури.

---

38. Елементи дійсних масивів $X=(x_1,x_2,...,x_n)$ і $Y=(y_1,y_2,...,y_n)$ визначають координати точок ламаної лінії. Вилучити із складу ламаної відрізок мінімальної довжини і відрізок максимальної довжини. Визначити, як при цьому змінилася загальна довжина ламаної лінії і середня довжина її відрізків.

Вказівка. Для вилучення першого або останнього відрізків досить вилучити відповідно першу або останню точку ламаної лінії. Для вилучення інших відрізків потрібно виконати паралельний перенiс частини точок.

Припустимо, що із складу ламаної потрібно вилучити відрізок $(x_k,y_k)-(x_{k+1},y_{k+1})$. У цьому випадку треба виконати наступні дії:

1) обчислити $dx=x_{k+1}-x_k;\quad dy=y_{k+1}-y_k;$

2) скорегувати координати точок

$$x_{k+2}:=x_{k+2}-dx;\quad x_{k+3}:=x_{k+3}-dx;\ ...;\ x_n:=x_n-dx;$$
$$y_{k+2}:=y_{k+2}-dy;\quad y_{k+3}:=y_{k+3}-dy;\ ...;\ y_n:=y_n-dy;$$

3) вилучити з масивів X і Y точку (x_{k+1},y_{k+1});

4) значення n зменшити на 1.

---

39. Якщо в дійсному масиві $X=(x_1,x_2,...,x_n)$ є серії, які складаються не менш ніж із трьох від'ємних елементів, то переставити елементи цих серій у зворотному порядку.

Примітка. Про методику пошуку серії див. п.2.

40. Перетворити масив $X=(x_1,x_2,...,x_n)$, розташувавши спочатку його від'ємні, а потім не від'ємні елементи, зберігши при цьому в групі невід'ємних елементів їх первісний відносний порядок, а в групі від'ємних елементів змінивши його на зворотний. Визначити, як при цьому змінилося положення мінімального елемента масиву *X*. Буферний масив не використовувати.

---

41. Розглядаючи кожну пару суміжних елементів заданої послідовності $X=(x_1,x_2,...,x_n)$ цілих додатних чисел, знайти таку з них, для якої найменше загальне кратне має максимальне значення.

Примітка. Найменше загальне кратне двох чисел *m* і *n*

$$k(m,n)=\frac{m\cdot n}{d(m,n)},$$

де $d(m,n)$ - найбільший загальний дільник, що обчислюється по алгоритму Евкліда.

Для змінних *m, n, k* рекомендується використати тип `longint`.

Для попередження переповнення результату обчислення значення *k* рекомендується робити в наступному порядку:

$$k=(n/d)\cdot n$$

Про методику пошуку серії див.п.2.

---

42. Задано два цілочисельних масиви $X=(x_1,x_2,...,x_n)$ і $Y=(y_1,y_2,...,y_m)$. До складу масиву *X* додатково включити ті елементи з масиву *Y*, які відсутні в масиві *X*. Визначити, як при цьому змінилося середнє арифметичне значення елементів масиву *X* (у відсотках).

---

43. У початковій частині заданого масиву $X=(x_1,x_2,...,x_n)$ розташувати нульові елементи, які входять до його складу, а потім у порядку зростання додатні елементи і в порядку убування від'ємні елементи. Визначити, як при цьому змінилося положення максимального й мінімального елементів масиву *X*. Буферний масив не використовувати.

Приклад.

Для масиву

X = (3, 8, -5, 0, 4, -1, 0, 7, -5, 0, 8, 10, -3)

одержимо

X = (0, 0, 0, 3, 4, 7, 8, 8, 10,,-1, -3, -5, -5).

---

44. Вилучити з масиву $X=(x_1,x_2,...,x_n)$ його максимальний і мінімальний елементи, після чого переставити елементи, які залишилися, у зворотному порядку. Визначити, як змінилося середнє арифметичне значення елементів масиву *X* після його перетворення (у відсотках). Буферний масив не використовувати.

---

45. Задано дійсні масиви $X=(x_1,x_2,...,x_n)$ і $Y=(y_1,y_2,...,y_m)$. Нехай масив *X* має k_1 додатних елементів, а масив *Y* - k_2 від'ємних елементів. Обміняти місцями $k=\min(k_1,k_2)$ додатних елементів масиву *X* с від'ємними елементами масиву *Y* (у порядку їхнього проходження в масивах *X* і *Y*). Врахувати, що в окремому випадку може бути *k* = 0.

---

46. Елементи масиву $Y=(y_1,y_2,...,y_n)$ - це ординати точок ламаної лінії в рівновіддалених вузлах по осі *X*. Виконати згладжування ламаної лінії по формулі

$y'_i = \frac{y_{i-1} + 2y_i + y_{i+1}}{4}$, $(i = 2,3,...,n-1)$; $y'_1 = \frac{y_1 + y_2}{2}$; $y'_n = \frac{y_{n-1} + y_n}{2}$, де y'_i- ординати перетвореної лінії. Визначити, наскільки змінилося середнє відхилення зубців ламаної щодо сусідніх точок.

Примітка. Вершини зубців ламаної - це точки, для яких виконується відношення $y_{i-1} < y_i > y_{i+1}$. Відхилення зубця щодо сусідніх точок $h = y_i - \frac{y_{i-1} + y_{i+1}}{2}$.

Зміна y_1 виробляється, якщо $y_1 > y_2$; аналогічна зміна y_n виконується, якщо $y_n > y_{n-1}$; для інших ординат – якщо $y_i > y_{i-1}$ і одночасно $y_i > y_{i+1}$.

---

47. У масиві $X = (x_1, x_2, ..., x_n)$ визначити положення й розмір останньої серії додатних непарних елементів, до складу якої входить від двох до п'яти елементів, після чого вилучити цю серію із складу масиву.

---

48. Вилучити із цілочисельного масиву $X = (x_1, x_2, ..., x_n)$ нульові елементи й переставити у зворотному порядку елементи, які залишилися. Визначити, як змінилися при цьому значення й положення мінімального $x_{\min}$ і максимального $x_{\max}$ елементів даного масиву, а також його середнє арифметичне значення S. Зміну значення параметра *S* виразити у відсотках. Буферний масив не використовувати.

---

49. Задано цілочисельний масив $X = (x_1, x_2, ..., x_n)$, у якому можуть бути однакові числа. Знайти найближчу від початку масиву пару однакових чисел, різниця індексів яких мінімальна, після чого вилучити ці числа з масиву.

Примітка. Цілком очевидно, що при виявленні двох суміжних однакових чисел мінімальна різниця індексів стає рівній одиниці. Тому при виявленні зазначеної ситуації подальший пошук заданих чисел потрібно припинити.

---

50. Задано два масиви $X = (x_1, x_2, ..., x_n)$ і $Y = (y_1, y_2, ..., y_m)$, до складу яких входять натуральні числа, причому в кожному із цих масивів немає повторюваних елементів. Сформувати масив *Z*, включивши в нього ті елементи з масиву *X*, які відсутні в масиві *Y*. Числа, перенесені в масив *Z*, вилучити з масиву *X*.

Приклад. Нехай ми маємо

X = (4, 8, 5, 7, 1`0, 18, 11, 14), n = 8

і Y = (6, 3, 7, 4, 9, 12, 16, 14, 11), m = 9.

Тоді одержимо

Z = (8, 5, 10, 18, 15) і X = (4, 7, 11).

К о н т р о л ь н а р о б о т а № 2

О Б Р О Б К А Т Е К С Т У

З а г а л ь н і м е т о д и ч н і в к а з і в к и

Мета роботи - практично освоїти методи розробки алгоритмів і їхньої програмної реалізації при обробці текстів, заданих у вигляді масиву рядків.

У всіх завданнях, якщо це не обговорено додатково, виконується обробка сторінки тексту, кількість рядків якої не перевищує 50, а кількість позицій у рядку не перевищує

заданого значення (наприклад, 66). Слова в тексті розділяються між собою одним або декількома пропусками (або іншими роздільниками, якщо це зазначено в завданні). Перенос слів з одного рядка на інший, як правило, не застосовується (використання переносу слів помітно ускладнює програму обробки тексту).

У контрольній роботі передбачається, що вхідний текст записаний у зовнішньому файлі. Після уведення з файлу він відображається у вигляді масиву рядків. Перетворений відповідно до умови завдання текст розміщується в тім же масиві рядків, що й вхідний (або в додатковому масиві рядків, якщо це потрібно по алгоритму рішення завдання), після чого він записується у вихідний файл. У деяких випадках оброблені фрагменти вхідного тексту безпосередньо пересилаються у вихідний файл, без утворення масиву рядків. Якщо за умовою завдання потрібно формувати список будь-яких елементів вхідного тексту (слів, констант і т.п.), то елементи списку у вихідному файлі, як правило, повинні розділятися комами.

Якщо в процесі перетворення вхідного тексту змінюються довжини його рядків, то перед записом тексту у вихідний файл потрібно виконати вирівнювання рядків тексту шляхом переносу у вільну праву частину розглянутого рядка деякої початкової частини наступного рядка. Оскільки в завданнях до даної лабораторної роботи перенос слів не використовується, то в кожному рядку тексту повинна бути розміщена максимально можлива ціла кількість слів. Зміна проміжку між словами для повного вирівнювання рядків тексту не виконується, якщо це спеціально не зазначено в завданні.

У вихідний файл виводяться також всі інші результати обробки вхідного тексту. Друк результатів рішення завдання повинен виконуватися шляхом читання вихідного файлу й виводу прочитаних рядків на принтер і (або) екран.

У кожному завданні вхідний файл повинен бути описаний як текстовий, а вихідний – як типізований, компонентами якого є рядки заданої довжини (наприклад, **`file of string`**`[66])`**.**

Якщо в завданні результати роботи програми безпосередньо записуються у вихідний файл, без формування перетвореного масиву рядків, то перед записом у цей файл, як правило, повинне бути забезпечене вирівнювання тексту по правій границі.

Пояснювальна записка по контрольній роботі № 2 – див. методичні вказівки до контрольної роботи №1.

Операції, процедури і функції для обробки рядків

Стосовно рядків може бути використана лише одна операція – операція зчеплення (конкатенації), що позначається символом ‘+’. Якщо змінні `S1 i S2` оголошені як рядки, то вираз `S1+S2` означає, що до кінця рядка `S1` дописується рядок `S2`. При цьому автоматично формується відповідне значення довжини об'єднаного рядка, що записується в її нульовий байт.

Приклад 1.

```
Var  S1,S2 : string[10];
        S3 : string[15];
        S4 : string;
Begin
  S1:='0123456789'; S2:='abcdefgh';
  S3:=S1+S2; S4:=S1+S2;
  Writeln('S3=',S3,'   S4=',S4);
  Буде надруковано:
    S3=0123456789abcde       { поточна довжина S3 15 символів }
    S4=0123456789abcdefgh    { поточна довжина S4 18 символів }
```

Припустимо, що рядок S4 треба доповнити символами '*' до загальної довжини 25. Це можна зробити різними способами, два з яких наведені в прикладах 2 і 3.

Приклад 2.

```
Var  S4 : string;
Begin
  S4:='0123456789abcdefgh';
  While ord(S4[0])<=25 do      { ord(S4[0]) – поточна довжина }
    S4:=S4+'*';                { рядка S4 }
  Writeln('   S4 = ',S4);
```

Буде надруковано:

```
 S4 = 0123456789abcdefgh*******
```

Приклад 3.

```
Var  i : byte;
    S4 : string;
Begin
  S4:='0123456789abcdefgh';
  For i:=ord(S4[0])+1 to 25 do
    S4[i]:='*';
  Writeln('   S4 = ',S4);
```

Буде надруковано:

```
 S4 = 0123456789abcdefgh
```

У прикладі 3 байти 21..25 рядка насправді будуть заповнені символом '*', але поточна довжина рядка, на відміну від приклада 2, при цьому не змінюється. Це пояснюється тим, що в прикладі 2 обробляється в цілому рядок, а в прикладі 3 – лише окремі його символи. Щоб результат обробки в прикладі 3 був таким же, як і в прикладі 2, потрібно після оператора **For** записати оператор S4[0]:=chr(25) (нульовий байт рядка, як і інші його байти, сприймається в програмі як символ; приведений вище оператор присвоювання заносить у нульовий байт рядка S4 символ, що відповідає порядковому номеру 25 таблиці ASCII).

Для рядків припустимі всі операції відношення. Порівняння рядків виконується зліва направо по одному байту доти, поки не будуть виявлені різні байти. Нехай у цих байтах розміщені символи ch1 і ch2. Тоді

ch1 < ch2, якщо ord(ch1) < ord(ch2).

Якщо поточні довжини порівнюваних рядків неоднаковs, то більш короткий рядок доповнюється символом chr(0), що менше будь-якого іншого символу таблиці ASCII (символ chr(0) має порядковий номер 0 у таблиці ASCII й умовно позначається Null; графічного зображення цей символ не має).

Приклад 4.

```
Var  b1,b2,b3 : boolean;
Begin
  b1:='abcdefgh'<'abcdxyzu';
  b2:='300'>'30'
  b3:=' '='1234567';
```

Тут маємо b1 = true; b2 = false; b3 = false.

Великі і малі букви в тексті вважаються різними. Зокрема, ‘d'≠ ‘D', тому що ord(‘d')≠ ord(‘D').

Для рядків визначено ряд процедур і функцій.

Процедура **Delete(S,Start,n)** виконує вилучення n символів з рядка S, починаючи з позиції Start.

Приклад 5.

```
Var  S : string[15];
Begin
  S:='0123456789';
  Delete(S,5,3);
```

Результат: S = ‘0123789’

Поточна довжина рядка S: вхідна 10, кінцева 7.

Процедура **Insert(S1,S2,Start)** - це вставка рядка S1 у рядок S2, починаючи з позиції Start.

Приклад 6.

```
Var  S1,S2,S3 : string[10];
Begin
  S1:='xyz'; S2:='01234'; S3:='0123456789';
  Insert(S1,S2,3); Insert(S1,S3,3);
```

Одержимо:

S2 = '01xyz234'; S3 = '01xyz23456';

Процедура **Str(X,S)** виконує перетворення числового значення арифметичного виразу X і розміщення результату в рядку S. Після виразу X можна записувати формат, аналогічний формату виводу в процедурах Write, Writeln.

Примітка. Виразом, зокрема, може бути константа, змінна, функція.

Приклад 7.

```
Var  x,y,z : real;
     k : integer;
     S1,S2,S3,S4 : string[10];
Begin
  x:=15.6789; y:=7; z:=123456789.123456789; k:=-789;
  Str(x:7:2,S1); Str(y:10,S2);
  Str(z:20:8,S3); Str(k:8,S4);
```

Одержимо:

S1 = ' 15.68'; S2 = ' 7.000E+00';

S3 = ‘123456789.’ (*дробова частина числа z не помістилася в S3*);

S4 = ' -789';

Для порівняння із процедурою Str розглянемо роботу процедури Write при виводі числа в текстовий файл (на екран, принтер, диск). При розміщенні числа на екрані

або на папері кожна цифра (символ) числа займає окрему позицію. Отже, число при розміщенні його в текстовому файлі - це рядок.

Робота процедури `Write` розділяється на два етапи:

- перетворення числової змінної з її внутрішньої машинної форми (відповідно до опису змінної в розділі **Var**) у рядок;
- вывод рядка на зовнішній носій інформації.

Перший етап роботи процедури `Write` еквівалентний роботі процедури `Str`.

Процедура **Val(S,X,Code)** виконує перетворення рядка `S` у число `X`. При цьому в рядку `S` не повинно бути пропусків між цифрами числа, а форма запису символів повинна відповідати правилам запису числових констант. Змінна `Code` типу `integer` – це код помилки. Якщо перетворення завершилося успішно, то `Code=0`. У противному випадку значення `Code` визначає позицію невірного символу, а змінної `X` при цьому буде привласнене нульове значення.

Приклад 8.

```
Var  x1,x2,x3 : real;
     k1,k2,Code1,Code2,
     Code3,Code4,Code5 : integer;
     S1,S2,S3,S4,S5 : string[10];
Begin
  S1:='  15.48'; S2:='  15,48'; S3:='  15.ab';
  S4:='  678'; S5:='  6 78';
  Val(S1,x1,Code1); Val(S2,x2,Code2);
  Val(S3,x3,Code3); Val(S4,k1,Code4);
  Val(S5,k2,Code5);
  Writeln('x1,Code1 = ',x1,'  ',Code1);
  Writeln('x2,Code2 = ',x2,'  ',Code2);
  Writeln('x3,Code3 = ',x3,'  ',Code3);
  Writeln('k1,Code4 = ',k1,'  ',Code4);
  Writeln('k2,Code5 = ',k2,'  ',Code5);
```

Буде надруковано:

```
x1,Code1 = 1.5480000000E+01  0
x2,Code2 = 0.0000000000E+00  5
x3,Code3 = 0.0000000000E+00  6
k1,Code4 = 678  0
k2,Code5 = 0  4
```

Для порівняння із процедурою `Val` розглянемо роботу процедури `Read` при уведенні числа з текстового файлу (із клавіатури або диска). Число в текстовому файлі має вигляд рядка, тому що кожна цифра (символ) числа займає окрему позицію на зовнішньому носії інформації. Роботу процедури `Read`, як і процедури `Write`, можна розділити на два етапи:

- перетворення рядка в значення числової змінної відповідно до її опису в розділі **Var**;
- пересилка значення змінної в задане поле пам'яті.

Перший етап роботи процедури `Read` еквівалентний роботі процедури `Val`.

Функція **Copy(S,Start,n):string** – це виділення з рядка `S` підрядка довжиною `n` байт, починаючи з позиції `Start`.

Приклад 9.

```
Var  S1,S2 : string;
Begin
  S1:='0123456789'; S2:=Copy(S1,3,4);
```

Одержимо:

```
S2 = '2345';
```

Функція **Concat(S1,S2,…,Sn):string**–це конкатенація (зчеплення) рядків `S1, S2,...,Sn`. Дія функції `Concat` еквівалентно послідовному виконанню операцій зчеплення `S1+S2+...+Sn`.

Функція **Length(S):byte** визначає поточну довжину рядка `S`. Дія цієї функції еквівалентно обчисленню виразів `ord(S[0])` або `byte(S[0])`.

Функція **Pos(S1,S2):byte** установлює позицію, у якій відзначається перша поява в рядку `S2` підрядка `S1`. Якщо `S1` не міститься в `S2`, то вихідне значення функції `Pos` дорівнює нулю.

Приклад 10.

```
Var  k1,k2,k3 : byte;
     S1,S2,S3 : string;
Begin
  S1:='abcdefghijklmnopq'; S2:='ghi'; S3:='0123';
  k1:=Pos(S2,S1); k2:=Pos(S3,S1); k3:=Pos('k',S1);
```

Одержимо:

```
k1 = 7;  k2 = 0;  k3 = 11
```

Функція **Upcase(ch):char** виконує перетворення малої латинської букви в велику. Символи `ch` поза діапазоном `'a'..'z'` залишаються без зміни.

Приклад 11.

```
Var  i : byte;
     S : string;
Begin
  S:='abc0123KLMN+фбщг-Uvwxyz';
  For i:=1 to length(S) do
    S[i]:=Upcase(S[i]);
```

Одержимо:

```
S = 'ABC0123KLMN+фбщг-UVWXYZ';
```

Для ілюстрації використання вищевказаних процедур і функцій розглянемо кілька прикладів.

Приклад 12.

У рядку задано текст. Слова тексту розділені між собою одним або декількома пропусками. На початку й наприкінці рядка також можуть бути пропуски. Стиснути текст, вилучивши з нього пропуски.

Розглянемо три варіанти програми.

```
Program Example12a;
Var  S : string;
     i : byte;
Begin
  Readln(S);
  For i:=1 to length(S) do
    If S[i]=' ' then
      Delete(S,i,1);
  Writeln(S);
End.
```

Із приводу програми Example12a можна зробити наступні зауваження:

1) Початкове й кінцеве значення параметра циклу **For** обчислюються до початку циклу й в процесі його виконання не змінюються. Отже, зменшення довжини рядка length(S) при S[i]=' ' не відображається на кількості повторень циклу по параметру i. Це приводить до надлишкової роботи програми, але не відбивається на правильності рішення завдання.

2) Припустимо, що в рядку S є пропуски, що йдуть підряд: S[k] = ' ' s S[k+1] =' '. При i=k буде вилучений елемент S[k], а на його місце буде пересланий елемент S[k+1], що також містить пропуск. Оскільки при новому повторенні циклу **For** параметр циклу приймає значення k+1, те новий елемент S[k], хоча він і містить у собі пропуск, не буде аналізуватися повторно і, як наслідок, не буде вилучений з масиву. Тому при наявності в масиві підряд розташованих пропусків програма Example12a буде працювати невірно.

Для правильного рішення поставленого завдання потрібно, щоб програма після вилучення елемента S[k]=' ' повторно аналізувала цей же елемент і переходила до розгляду елемента S[k+1] лише при S[k]<>' '. Для цього потрібно в програмі замість циклу **For** використати цикл **While.**

```
Program Example12b;
Var  S : string;
     i : byte;
Begin
  Readln(S); i:=1;
  While i<= length(S) do
    If S[i]=' ' then
      Delete(S,i,1)
    Else
      Inc(i);
  Writeln(S);
End.
```

Правильна обробка рядка S буде зроблена також, якщо в циклі **For** виконувати перегляд символів рядка не зліва направо, а справа наліво (програма Example12c).

```
Program Example12c;
Var  S : string;
     i : byte;
Begin
  Readln(S);
  For i:=length(S) downto 1 do
```

```
    If S[i]=' ' then
      Delete(S,i,1);
  Writeln(S);
End.
```

Приклад 13. У рядку задано текст. Слова тексту розділені між собою одним або кількома пропусками. На початку й наприкінці рядка також можуть бути пропуски. Визначити кількість слів у тексті й середню кількість букв у слові.

Припустимо, що рядок `S` має такий вигляд (символ '-' тут умовно використаний як пропуск):

```
--abcde----fghijklmn----n-opq-xy-
```

Ознакою початку слова є символ, відмінний від пропуска (непропуск), ознакою закінчення слова – найближчий до нього пропуск. Будемо привласнювати номера позицій, які визначають положення слова в рядку, змінним `k1` і `k2`:

```
--abcde----fghijklmn----n-opq-xy-
  ↑    ↑    ↑         ↑
  k1   k2   k1        k2
```

Отже, при аналізі рядка в програмі потрібно багаторазово визначати місце розташування слова або, інакше кажучи, позицію найближчого пропуска або непропуска, починаючи із заданої позиції `k`. Для виконання такої роботи доцільно написати окремі підпрограми, оформивши їх у вигляді функції, оскільки виходом підпрограми є одне просте значення - позиція пропуска або непропуска.

Пошук найближчого пропуска:

```
Function Space(S:string; k:byte):byte;
Var  i : byte;
Begin
  Space:=0;
  For i:=k to length(S) do
    If S[i]=' ' then
      Begin
        Space:=i; Exit
      End;
End { Space };
```

Якщо в рядку `S,` починаючи з позиції `k,` пропуск не виявлено, то вихідне значення `Space=0.`

Пошук найближчого непропуска:

```
Function NotSpace(S:string; k:byte):byte;
Var  i : byte;
Begin
  NotSpace:=0;
  For i:=k to length(S) do
```

```
    If S[i]<>' ' then
      Begin
        NotSpace:=i; Exit
      End;
End { NotSpace };
```

Будемо вважати, що в програмі Example13, що реалізує рішення поставленого завдання, містяться функції Space і NotSpace.

```
Program Example13;
Uses Crt;
Var  S : string;       { оброблюваний рядок }
    NumWords,          { кількість слів }
    NumLetters,        { кількість букв }
    k1,k2,             { ліва й права границі слова }
    len : byte;        { довжина слова }
    MiddleLet : real;  { середня кількість букв у слові }
    Cond : boolean;    { змінна для керування циклом }
{ ------------------------------------- }
Функції  Space й NotSpace
{ ------------------------------------- }
Begin
  ClrScr;
  Readln(S); Writeln('S = ',S);
  NumWords:=0; NumLetters:=0;
  MiddleLet:=0;
  k2:=0; Cond:=true;
  While Cond do
    Begin
      k1:=NotSpace(S,k2+1);
      If k1=0 then
        Cond:=false
      Else
        Begin
          k2:=Space(S,k1+1);
          If k2=0 then
            Begin
              k2:=length(S)+1; Cond:=false
            End;
          len:=k2-k1;
          Inc(NumWords); Inc(NumLetters,len);
        End;
    End;
  If NumWords>0 then        { блокування ділення на нуль }
    MiddleLet:=NumLetters/NumWords;
  Writeln('Кількість слів = ',NumWords,
          '   Кількість букв = ',NumLetters);
  Writeln('Середня довжина слова = ',MiddleLet:6:1);
End.
```

У програмі здійснюється послідовне виділення слів тексту шляхом визначення границь k1 і k2. Стартове значення змінної k2 приймається рівним нулю. Цикл пошуку слів триває доти, поки є істинним значення змінної Cond.

У кожному циклі пошуку визначаються значення змінних k1 і k2. Значення k1 – це позиція найближчого непропуска, починаючи з байта k2+1 (у першому циклі k2+1=1). Якщо при звертанні до функції NotSpace отримано k1=0, то це означає, що до кінця рядка більше слів не виявлено. Тоді змінній Cond привласнюється значення false, що веде до припинення циклу пошуку.

Якщо k1<>0, то визначається значення k2, тобто номер позиції найближчого пропуска, починаючи з байта k1+1. Якщо при звертанні до функції Space отримано k2=0, то це означає, що до кінця рядка немає більше пропусків. Тоді правою границею слова вважається байт length(S)+1. Разом з тим значення k2=0 указує, що в тексті знайдене останнє слово. Тоді змінній Cond привласнюється значення false, що після обробки цього слова викличе припинення роботи циклу пошуку.

Приклад 14. У рядку розміщуються слова, які складаються з латинських букв, і цілі десяткові числа. Перед числом може стояти знак '+' або '-'. Слова й числа розділені в тексті пропусками. Визначити:

- кількість чисел у тексті;
- загальну суму десяткових цифр, які містяться в цих числах.

Пошук числа в тексті виконується аналогічно пошуку слова в попередньому прикладі, але ознакою початку числа є цифра або символи '+', '-'. Тому замість функції NotSpace у цьому випадку для пошуку числа будемо використовувати функцію Number.

```
Function Number(S:string; k:byte):byte;
Const  Sig = '+-0123456789';
Var  i : byte;
Begin
  Number:=0;
  For i:=k to length(S) do
    If Pos(S[i],Sig)>0 then
      Begin
        Number:=i; Exit
      End;
End { Number };
```

Будемо вважати, що до складу програми Example14 входять функції Space й Number.

```
Program Example14;
Uses Crt;
Var  S : string;       { оброблюваний рядок }
     i,                { параметр циклу }
     NumNumbers,       { кількість чисел }
     k1,k2,            { ліва й права границі числа }
     Dig : byte;       { числове значення цифри }
     Code : integer;   { код перетворення }
     Sum : longint;    { сума цифр }
     Cond : boolean;   { змінна для керування циклом }
{ ---------------------------------------- }
Функції  Space й Number
{ ---------------------------------------- }
Begin
```

```
  ClrScr;
  Readln(S); Writeln('S = ',S);
  NumNumbers:=0; Sum:=0;
  k2:=0; Cond:=true;
  While Cond do
    Begin
      k1:=Number(S,k2+1);
      If k1=0 then
        Cond:=false
      Else
        Begin
          k2:=Space(S,k1+1);
          If k2=0 then
            Begin
              k2:=length(S)+1; Cond:=false
            End;
          Inc(NumNumbers);
          If (S[k1]='+') or (S[k1]='-') then
            Inc(k1);
          For i:=k1 to k2-1 do
            Begin
              Val(S[i],Dig,Code);
              Inc(Sum,Dig);
            End;
        End;
    End;
  Writeln('Кількість чисел = ',NumNumbers,
          '   Сума цифр = ',Sum);
End.
```

Приклад 15. До складу тексту входять слова й шістнадцятеричні числа, ціла й дробова частини яких розділені крапкою. Ознакою 16-го числа є префікс '$'. Перед числом може стояти знак '+' або '-'. Слова й числа розділені між собою одним або декількома пропусками. Перетворити шістнадцятеричні числа в десяткову систему числення, зображуючи їх у вигляді рядка. У дробовій частині десяткового числа записувати не більш ніж 10 цифр. Незначні нулі, якщо вони є в дробовій частині числа, вилучити із складу рядка.

Принцип пошуку 16-го числа в основному аналогічний прикладу 14, але у функції Number ознакою початку числа є символ '$'. Границі числа, як і раніше, визначаються змінними k1 і k2, при цьому знак числа, якщо він є, знаходиться в позиції k1-1.

Далі будемо виконувати безпосереднє перетворення знайденого числа, записаного в рядок Sb (без символу '$'):

```
Sb:=Copy(S,k1+1,k2-k1-1)
```

Позначимо через m довжину рядка Sb, через k – позицію крапки, що розділяє цілу й дробову частини числа:

```
m:=length(Sb);  k:=Pos('.',Sb)
```

Тоді цифри цілої частини займають у рядку Sb позиції з 1 до k-1, а дробової – з k+1 до m.

Нехай у цілій частині 16-го числа міститься n цифр. Тоді це число можна зобразити в наступному виді:

$$a_1 \cdot 16^{n-1} + a_2 \cdot 16^{n-2} + ... + a_{n-1} \cdot 16^1 + a_n \cdot 16^0$$

або за схемою Горнера:

$$(...(a_1 \cdot 16 + a_2) \cdot 16 + a_3) \cdot 16 + ... + a_{n-1}) \cdot 16 + a_n \ ,$$

де a_i, $i = 1..n$ - шістнадцятеричні цифри.

У наведених нижче фрагментах програми передбачається, що вхідний текст розміщується в рядку S, з якого розглянутим раніше методом визначається рядок Sb, що включає в себе шістнадцятеричн число.

Програма перетворення цілої частини числа:

```
Const  Digs = '0123456789ABCDEF'; { шістнадцятеричні }
                                  { цифри }
Var  Nb1 : longint;  { ціла частина десяткового числа }
     Nb2 : real;     { дробова частина такого числа }
     i,p,k,m : byte;
     S,              { вхідний рядок }
     Sb,             { шістнадцятеричне число як рядок }
     St1,            { ціла частина 10-го числа як рядок }
     St2 : string;   { те ж для дробової частини }
.................
  { Визначення числа Sb з вхідного рядка S }
  m:=length(Sb); k:=Pos('.',Sb);
  Nb1:=0;
  For i:=1 to k-1 do
    Begin
      p:=Pos(Sb[i],Digs)-1;  { значення 16-ої цифри }
      Nb1:=Nb1*16+p
    End;
  Str(Nb1,St1);
  If S[k1-1]='+' then        { додавання знака числа }
    St1:='+'+St1
  Else
    If S[k1-1]='-' then
      St1:='-'+St1;
```

Припустимо, що Sb = '1AE.5A'. Тоді
k = 4; m = 6; Nb1 = 430; St1 = '430'.

Нехай тепер у дробовій частині числа розміщується n цифр. Це число можна зобразити в наступному виді:

$$b_1 \cdot 16^{-1} + b_2 \cdot 16^{-2} + b_3 \cdot 16^{-3} + ... + b_n \cdot 16^{-n} \ ,$$

де b_i, $i = 1..n$ - цифри дробової частини числа.

Запишемо цей вираз у зворотному порядку:

$$b_n \cdot 16^{-n} + b_{n-1} \cdot 16^{-(n-1)} + b_{n-2} \cdot 16^{-(n-2)} + ... + b_2 \cdot 16^{-2} + b_1 \cdot 16^{-1\cdot} \ .$$

Тоді за схемою Горнера

$$(...((b_n \cdot 16^{-1} + b_{n-1}) \cdot 16^{-1} + b_{n-2}) \cdot 16^{-1} + ... + b_2) \cdot 16^{-1} + b_1) \cdot 16^{-1}$$

або

$$(...((b_n / 16 + b_{n-1}) / 16 + b_{n-2}) / 16 + ... + b_2) / 16 + b_1) / 16$$

Вважаючи, що в рядку Sb дробова частина числа займає позиції з k+1 до m, одержимо представлену нижче програму перетворення дробової частини числа:

```
Nb2:=0;
For i:=m downto k+1 do
  Begin
    p:=Pos(Sb[i],Digs)-1;
    Nb2:=(Nb2+p)/16;
  End;
Str(Nb2:12:10,St2);                { 1 }
While St2[length(St2)]='0' do
  Delete(St2,length(St2),1);       { 2 }
Delete(St2,1,1);                   { 3 }
```

Для Sb = '1AE.5A' одержимо:
- після { 1 } Nb2 = 3.5156250000E-01; St2 = '0.3515625000';
- після { 2 } St2 = '0.3515625';
- після { 3 } St2 = '.3515625'.

Об'єднання цілої і дробової частин числа:

```
St1:=St1+St2
```

Остаточний результат: St1 = '430.3515625'.

Приклад 16. Пошук у рядку пари слів, які відповідають заданій ознаці.

Ознаки можуть бути різними (пари одинакових слів, пари слів зі зворотним розташуванням букв та ін.). Принциповим тут є організація циклу пошуку.

Нехай у рядку міститься *n* слів. Відповідно до умови завдання потрібно порівнювати всі пари слів:

```
1 - 2    1 - 3    1 - 4    1 - 5    1 - n
         2 - 3    2 - 4    2 - 5    2 - n
                  3 - 4    3 - 5    3 - n
                  ......................
                                (n - 1) - n
```

Якби мова йшла про порівняння елементів масиву, цикл пошуку можна було б легко організувати за допомогою оператора **For:**

```
For i:=1 to n-1 do
  For j:=i+1 to n do
    If x[i]=x[j] then ...
```

Однак для рядка заздалегідь невідома кількість записаних у ньому слів. Тому замість оператора **For** у цьому випадку потрібно використати оператор **While**. Суть організації циклу пошуку можна проілюструвати таким фрагментом програми:

```
Cond1:=true; k2:=0;
While Cond1 do
  Begin
    k1:=NotSpace(S,k2+1); { Виділення першого з }
    If k1=0 then          { пари порівнюваних }
```

```
      Cond1:=false          { слів }
    Else
      Begin
        k2:=Space(S,k1+1);
        If k2=0 then        { Це останнє слово - після }
          Cond1:=false      { його немає інших слів для }
        Else                { порівняння }
          Begin
            Sb1:=Copy(S,k1,k2-k1);
            Cond2:=true; k4:=k2+1;
            While Cond2 do
              Begin
                k3:=NotSpace(S,k4+1); { Виділення другого }
                If k3=0 then          { з пари порівнюва- }
                  Cond2:=false        { их слів }
                Else
                  Begin
                    k4:=Space(S,k3+1);
                    If k4=0 then
                      Begin
                        k4:=length(S)+1; Cond2:=false
                      End;
                    Sb2:=Copy(S,k3,k4-k3);
                    { Аналіз пари слів Sb1 й Sb2 }
                  End;
              End;
          End;
      End;
  End;
```

Приклад виконання завдання

Як приклад виконання завдання по контрольній роботі №2 наведені рішення двох завдань.

Умова завдання 1. До складу вхідного тексту входять слова, які складаються з латинських букв, цілі десяткові числа й роздільники. Скласти таблицю, у якій для кожної латинської букви поставити у відповідність слово, у якому ця буква вперше зустрічається, і номер даного слова; після цього ті слова, які занесені в таблицю, вилучити із вхідного тексту.

Рішення завдання 1 наведено в представленій нижче програмі `Cont4_1.`

Вхідний текст розміщується в текстовому файлі `FileText`, якому за допомогою передописаної процедури `Assign` ставиться у відповідність цілком певне ім'я зовнішнього файлу.

Робота будь-якої програми, у тому числі й `Cont4_1`, повинна бути перевірена на системі тестів, у яких записані різні комбінації вхідних даних. Якщо в процедурі `Assign` указати конкретне ім'я зовнішнього файлу, наприклад, `'Intext.txt'`, тоді при переході від одного тесту до іншого необхідно попередньо набирати із клавіатури новий уміст файлу `'Intext.txt'` або пересилати у файл `'Intext.txt'` інший файл, що містить необхідний тест. Це зв'язано зі значними незручностями для користувача.

Більше прийнятним рішенням є така організація роботи програми, при якій користувач міг би вказувати по запиті програми цілком або частково ім'я зовнішнього файлу, у якому перебувають необхідні вхідні дані.

Процедура `Assign(F,S)` містить два параметри: ім'я внутрішнього файлу `F`, описане в розділі **`Var`**, і ім'я зовнішнього файлу у вигляді рядка `S`. Рядок `S` може бути константою або змінною.

Ім'я зовнішнього вхідного файлу в програмі `Cont4_1` формується в рядку `NameInFile` і може приймати значення `InText0.txt, InText1.txt,..., InText9.txt`, тобто всього допускається 10 файлів, які містять тестові вхідні дані. Імена файлів відрізняються лише однією цифрою, що умовно можна вважати номером вхідного файлу. Цей номер користувач повідомляє із клавіатури після відповідного запиту програми. Уведення вхідного тексту здійснюється з файлу `InText.txt`, де `P` – номер вхідного файлу. Змінна `n` у програмі `Cont4_1` визначає кількість рядків уведеного тексту. Уведений текст розміщається в масиві рядків `St` і виводиться на екран дисплея, а також, якщо це потрібно, друкується на принтері.

Таблиця результатів складається із чотирьох стовпців: порядковий номер рядка таблиці; значення латинської букви; номер слова, у якому вперше зустрічається ця буква (масив `TabNumberWord`); значення цього слова (масив `TabWord`). Всім елементам масиву `TabNumberWord` задається початкове нульове значення, а елементам масиву рядків `TabWord` - початкове порожнє значення. Якщо при аналізі вхідного тексту не буде виявлена яка-небудь буква, то такі значення елементів масивів `TabNumberWord` і `TabWord` будуть свідчити про відсутність цієї букви.

У словах вхідного тексту можуть бути як великі, так і малі латинські букви, які в даному завданні вважаються рівноцінними. Для визначення приналежності чергового символу тексту до букв використовується рядок-константа `AlphaBet`, у якому розміщені лише великі латинські букви. Надалі при аналізі тексту для перетворення малих латинських букв у великі використовується функція `UpCase`.

Формування таблиці результатів і одночасно вилучення з тексту слів, у яких вперше зустрілася яка-небудь буква, здійснюється в процедурі `MakeTabResult`.

При аналізі тексту послідовно визначаються номер чергового слова `NumberWord` і кількість вперше зустрінутих букв `KolLetter`. Цим змінним до початку перегляду тексту привласнюється нульове значення.

Перегляд вхідного тексту здійснюється в циклі з параметром `i`, де `i` визначає номер рядка тексту, тобто індекс елемента масиву рядків `St`.

Визначення границь чергового слова в `i-ому` рядку тексту виконується в такий же спосіб, як це робиться в прикладах 13 і 14, але для індикації початку й кінця слова використовуються функції `SignBegin` і `SignEnd`. Відділення слова, що складається з латинських букв, від десяткового числа здійснюється по його першому символу. Після цього переглядаються всі букви слова і, якщо чергова буква зустрічається в тексті вперше (`TabNumberWord[k]=0`, де `k` – номер букви в рядку `AlphaBet`), те вона заноситься в таблицю результатів, а змінній `CondDelete` привласнюється значення `true`. В останньому випадку виконується вилучення даного слова з `i-oro` рядка тексту.

Наступний етап обробки тексту – вилучення надлишкових пропусків, що виконується процедурою `DeleteSpaces`. Тут здійснюється вилучення тих пропусків, після яких є який-небудь роздільник, в окремому випадку інший пропуск. Отже, у тексті зберігаються пропуски, які записані після інших роздільників, наприклад, після крапки або коми. Після обробки чергового рядка здійснюється вилучення пропусків на початку й наприкінці рядка (якщо вони є).

У процесі перегляду вхідного тексту з його складу була вилучена деяка кількість слів, що привело до скорочення довжини рядків тексту. Для вирівнювання цих рядків по

правій границі необхідно в кожен рядок, довжина якого менше оголошеної (у цьому випадку 66), перенести максимально можливу початкову частину наступного рядка. Природно, що при такому переносі цілі десяткові числа, що містяться в складі вхідного тексту, не повинні ділитися на дві частини. Слова, відповідно до загальних методичних вказівок, також не можна розділяти знаком переносу. Тому з наступного рядка треба взяти таку початкову частину, що закінчується роздільником, а не буквою або цифрою (тобто брати в цілому слова й числа).

Для визначення приналежності аналізованого символу до роздільників використається рядок-константа `Separs`, описана в розділі **`Const.`** У цьому рядку як роздільники перераховані пропуск, кома, крапка, крапка з комою, двокрапка. Для індикації початку й кінця чергового слова, як і в процедурі `MakeTabResult`, використовуються функції `SignBegin` і `SignEnd`.

В процесі вирівнювання тексту не виключена можливість того, що загальна кількість рядків `n` перетвореного тексту скоротиться, якщо всі символи останніх рядків будуть перенесені в попередні рядки. У зв'язку із цим для перебору рядків тексту в процесі його вирівнювання не можна застосовувати цикл **`For.`**

У процедурі `RightBorder` змінна `i` визначає номер рядка оброблюваного тексту, `is` – номер рядка нового (вирівняного) тексту. Перебір рядків здійснюється в циклі **`While`** під керуванням змінної `CondText`.

Для формування нового рядка використовується змінна `S1`, в яку до початку циклу перебору заноситься перший рядок оброблюваного текстового масиву `St.` У змінну `S2` заноситься черговий `i-ий` рядок оброблюваного тексту (початкове значення `i=2`). Якщо сумарна довжина `l1+l2` рядків `S1` і `S2` менше 66, то до рядка `S1` дописується вміст рядка `S2`, а значення змінної `i` збільшується на 1. Якщо при цьому має місце `i>n`, то керуючій змінній `CondText` привласнюється значення `false`, що веде до припинення циклу перебору рядків вхідного тексту.

При `l1+l2>66` здійснюється частковий перенос слів з рядка `S2` у рядок `S1`, що виконує цикл **`While`** під керуванням змінної `CondString`. Границі чергового слова в рядку `S2` визначають змінні `k1` і `k2`; значення `l=k2-k1` – це довжина слова. Якщо чергове слово може бути розміщене в рядку `S1` (`l1+l<=66`), то воно записується в `S1` і вилучається з рядка `S2`. Відношення `l1+l>66` означає, що рядок `S1` уже заповнений. У цьому випадку цикл перегляду рядка `S2` припиняє свою роботу, до значення змінної `is` додається 1, уміст рядка `S1` записується в рядок `St[is]` нового тексту, а в `S1` записується залишок рядка `S2`, після чого номер рядка `i` старого тексту збільшується на 1. При повторенні зовнішнього циклу в `S2` буде занесений черговий рядок перетвореного масиву `St.`

Після закінчення зовнішнього циклу **`While`** останньому `n-ому` рядку нового тексту привласнюється значення рядка `S1`.

Перетворений текст записується у вихідний типізований файл `FileOut`, після чого читається із цього файлу, виводиться на екран і, якщо це зазначено користувачем, на принтер.

Примітка. У програмі `Cont4_1` для збільшення й зменшення цілої змінної на величину, обумовлену константою або целочисленною змінною, використовуються процедури `Inc` (інкремент) і `Dec` (декремент).

```
Inc(i)      еквівалентно   i:=i+1;
Inc(i,5)    еквівалентно   i:=i+5;
Inc(i,k)    еквівалентно   i:=i+k;
Dec(i)      еквівалентно   i:=i-1;
```

```
Dec(i,5)    еквівалентно   i:=i-5;
Dec(i,k)    еквівалентно   i:=i-k.
```

Процедури Inc й Dec породжують оптимизований машинний код, що особливо корисно для великих програм, сприяючи при цьому зменшенню обсягу займаної пам'яті і збільшенню швидкодії програми.

```
Program Cont4_1;
Uses Crt,Printer;
Const
    Nmax= 50;       { макс. кількість эл-тов у масиві рядків St }
    Enter = 13;     { код клавіші Enter }
    Escape = 27;    { код клавіші Esc }
    PressKey = 'Натисніть клавішу Enter';
    AlphaBet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'; { лат. алфавіт }
    Separs = ' .,;:';         { список роздільників }
Type
    String66 = string[66];    { тип рядка тексту }
    StringAr = array[1..Nmax] of String66;
Var
    i,                      { параметр циклу }
    n,                      { кількість рядків у масиві St }
    NumberFile,             { номер вхідного файлу }
    KolLetter               { кількість відзначених букв }
          : byte;
    NumberWord
          : integer;        { поточний номер слова в тексті }
    IndPrinter : boolean;   { індикатор використання принтера }
    ch,                     { символ тексту }
    Reply : char;           { символ відповіді на запит програми }
    S1,S2,                  { рядки для перетворення тексту }
    Sw,                     { чергове слово тексту }
    NameInFile : String66;  { ім'я вхідного файлу }
    St : StringAr;          { масив рядків тексту }
    TabWord                 { масив слів таблиці результатів }
       : array[1..26] of String66;
    TabNumberWord           { номера слів у таблиці результатів }
       : array[1..26] of integer;
    FileText : text;        { вхідний файл }
    FileOut                 { вихідний файл }
             : file of String66;
{ ------------------------------------------------------------ }
Procedure WaitEnter;
{ Затримка виконання програми, поки не буде натиснута }
{ клавіша Enter                                        }
Var ch : char;
Begin
  Repeat
    ch:=ReadKey;
  Until ord(ch) = Enter;
End { WaitEnter };
{ ------------------------------------------------------------ }
Procedure PrintString(X,Y:integer; S:string);
```

```
{ Вивід рядка S у позицію X рядка екрана з номером Y }
Begin
  GotoXY(X,Y);
  Write(S);
End { PrintString };
{ ------------------------------------------------------------ }
Procedure PrintKeyAndWaitEnter;
{ Вивід рядка-константи PressKey у позицію 1 рядка екрана 25 }
{ і затримка виконання програми до натиску клавіші Enter      }
Begin
 PrintString(1,25,PressKey);
 WaitEnter;
 ClrScr;
End { PrintKeyAndWaitEnter };
{ ------------------------------------------------------------ }
Procedure ControlPageScreen(Var j,KeyExit:byte);
{ Контроль розміру сторінки на екрані }
Const LengthPage=23;    { кількість рядків на сторінці }
      S='Наступна сторінка - Enter, кінець перегляду - Esc';
Var ch : char;
Begin
  Inc(j); KeyExit:=0;
  If j=LengthPage then
    Begin
      j:=0;
      PrintString(1,25,S);
      Repeat
        ch:=ReadKey; KeyExit:=ord(ch);
      Until (KeyExit=Enter) or (KeyExit=Escape);
      ClrScr;
    End;
End { ControlPageScreen };
{ ------------------------------------------------------------ }
Procedure ScreenText(S:string);
{ Вивід на екран масиву рядків St }
Var  i : integer;
    j,KeyExit : byte;
Begin
  ClrScr; j:=0;
  Writeln(S);
  For i:=1 to n do
    Begin
      Writeln(St[i]);
      ControlPageScreen(j,KeyExit);
      If KeyExit=Escape then
        Exit;
    End;
  PrintKeyAndWaitEnter;
End { ScreenText };
{ ------------------------------------------------------------ }
Procedure PrinterText;
{ Друк на принтері вхідного тексту }
Var  i : integer;
```

```
Begin
  Writeln(Lst);
  Writeln(Lst,'              В Х І Д Н И Й   Т Е К С Т');
  For i:=1 to n do
    Writeln(Lst,St[i]);
End { PrinterText };
{ ------------------------------------------------------------ }
Procedure ScreenTabResult;
{ Вивід на екран таблиці результатів }
Var  i : integer;
    j,KeyExit : byte;
Begin
  ClrScr; j:=1;
  Writeln('               Таблиця результатів');
  For i:=1 to 26 do
    Begin
      Writeln(i:3,'   ',Copy(AlphaBet,i,1),'   ',
              'TabNumberWord[i]:3,'   ',TabWord[i]);
      ControlPageScreen(j,KeyExit);
      If KeyExit=Escape then
        Exit;
    End;
  PrintKeyAndWaitEnter;
End { ScreenTabResult };
{ ------------------------------------------------------------ }
Procedure ScreenOutFile;
{ Вивід на екран вихідного файлу }
Var j,KeyExit : byte;
Begin
  Seek(FileOut,0);
  ClrScr; j:=0;
  Writeln('              В И Х І Д Н И Й   Ф А Й Л');
  While not eof(FileOut) do
    Begin
      Read(FileOut,S1);
      Writeln(S1);
      ControlPageScreen(j,KeyExit);
      If KeyExit=Escape then
        Exit;
    End;
  PrintKeyAndWaitEnter;
End { ScreenOutFile };
{ ------------------------------------------------------------ }
Procedure PrinterOutFile;
{ Друк на принтері вихідного файлу }
Begin
  Seek(FileOut,0);
  Writeln(Lst); Writeln(Lst);
  Writeln(Lst,'              В И Х І Д Н и Й   Ф А Й Л');
  While not eof(FileOut) do
    Begin
      Read(FileOut,S1);
      Writeln(Lst,S1);
```

```
    End;
End { PrinterOutFile };
{ ------------------------------------------------------------ }
Function SignBegin(S:string66; k:byte):byte;
{ Пошук початку слова в рядку S }
Var i : byte;
Begin
  SignBegin:=0;
  For i:=k to length(S) do
    If Pos(S[i],Separs)=0 then
      Begin
        SignBegin:=i; Exit
      End;
End { SignBegin };
{ ------------------------------------------------------------ }
Function SignEnd(S:string66; k:byte):byte;
{ Пошук кінця слова в рядку S }
Var i : byte;
Begin
  SignEnd:=0;
  For i:=k to length(S) do
    If Pos(S[i],Separs)>0 then
      Begin
        SignEnd:=i; Exit
      End;
End { SignEnd };
{ ------------------------------------------------------------ }
Procedure MakeTabResult;
{ Формування таблиці результатів і вилучення відзначених слів }
Var  i,j,k,k1,k2 : byte;
     Cond,
     CondDelete : boolean;
Begin
  NumberWord:=0; KolLetter:=0;
  For i:=1 to n do
    Begin
      Cond:=true; k2:=0;
      While Cond do
        Begin
          k1:=SignBegin(St[i],k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=SignEnd(St[i],k1+1);
              If k2=0 then
                Begin
                  k2:=length(St[i])+1; Cond:=false;
                End;
              Sw:=Copy(St[i],k1,k2-k1);
              ch:=UpCase(Sw[1]);
              If Pos(ch,AlphaBet)>0 then
                Begin
```

```
                    Inc(NumberWord);
                    CondDelete:=false;
                    For j:=1 to length(Sw) do
                      Begin
                        ch:=UpCase(Sw[j]); k:=Pos(ch,AlphaBet);
                        If TabNumberWord[k]=0 then
                          Begin
                            TabNumberWord[k]:=NumberWord;
                            TabWord[k]:=Sw;
                            Inc(KolLetter);
                            CondDelete:=true;
                          End;
                      End;
                    If CondDelete then
                      Begin
                        Delete(St[i],k1,k2-k1);
                        k2:=k1;
                      End;
                    If KolLetter=26 then
                      Exit;
                  End;
              End;
          End;
      End;
End { MakeTabResult };
{ ------------------------------------------------------------ }
Procedure WriteTabResult;
{ Запис таблиці результатів у вихідний файл }
Var  i : byte;
Begin
  ScreenTabResult;
  S1:=' ';
  Write(FileOut,S1);
  S1:='            ТАБЛИЦЯ ВИХІДНИХ РЕЗУЛЬТАТІВ';
  Write(FileOut,S1);
  S1:='-------------------------------------------------------';
  Write(FileOut,S1);
  S2:=': N п/п : Буква : Номер слова :        Слово         :';
  Write(FileOut,S2); Write(FileOut,S1);
  For i:=1 to 26 do
    Begin
      Str(i:2,S1); Str(TabNumberWord[i]:4,S2);
      S1:='   '+S1+'       '+Copy(AlphaBet,i,1)+'      '+
                S2+'        '+TabWord[i];
      Write(FileOut,S1);
    End;
End { WriteTabResult };
{ ------------------------------------------------------------ }
Procedure DeleteSpaces;
{ Вилучення з тексту надлишкових пропусків }
Var  i,j : byte;
Begin
  For i:=1 to n do
```

```
    Begin
      S1:=St[i];
      j:=1;
      While j<length(S1) do
        If S1[j]=' ' then
          Begin
            ch:=S1[j+1];
            If Pos(ch,Separs)>0 then
              Delete(S1,j,1)
            Else
              Inc(j);
          End
        Else
         Inc(j);
      If S1[length(S1)]=' ' then
        Delete(S1,length(S1),1);
      If S1[1]=' ' then
        Delete(S1,1,1);
      St[i]:=S1;
    End;
End { DeleteSpaces };
{ ------------------------------------------------------------- }
Procedure RightBorder;
{ Вирівнювання скоректованого тексту по правій границі }
Var  is,l,l1,l2,
     k,k1,k2 : byte;
     CondText,CondString : boolean;
Begin
  If n<2 then
    Exit;
  S1:=St[1]; l1:=length(S1);
  i:=2; is:=0; CondText:=true;
  While CondText do
    Begin
      S2:=St[i]; l2:=length(S2);
      If (l1+l2)<66 then
        Begin
          S1:=S1+' '+S2; l1:=length(S1);
          Inc(i);
          If i>n then
            CondText:=false
        End
      Else
        Begin
          k:=66-l1;
          If k>0 then
            Begin
              CondString:=true;
              While CondString do
                Begin
                  k1:=SignBegin(S2,1);
                  k2:=SignEnd(S2,k1+1);
                  If S2[k2]<>' ' then
```

```
                    Inc(k2);
                  l:=k2-k1;
                  If l1+l<=66 then
                    Begin
                      Sw:=Copy(S2,k1,k2-k1);
                      S1:=S1+' '+Sw;
                      Delete(S2,1,k2-1);
                    End
                  Else
                    CondString:=false;
                End;
              Inc(is); St[is]:=S1;
              S1:=S2;
              If S1[1]=' ' then
                Delete(S1,1,1);
              l1:=length(S1);
              Inc(i);
              If i>n then
                CondText:=false;
            End;
        End;
    End;
  n:=is+1; St[n]:=S1;
End { RightBorder };
{ ------------------------------------------------------------ }

Begin

{ Установлення відповідності між внутрішнім }
{ і зовнішнім файлами }
  ClrScr;
  Writeln('Укажіть номер вхідного файлу (0..9) ');
  Read(NumberFile); Str(NumberFile:1,S1);
  NameInFile:='InText'+S1+'.txt';
  Assign(FileText,NameInFile);
  Assign(FileOut,'OutText.dat');

{ Відкриття використовуваних файлів }
  Reset(FileText);
  Rewrite(FileOut);

{ Запит про використання принтера }
  IndPrinter:=false;
  Writeln(' Чи буде використаний принтер ?  (Так,Ні)');
  Reply:=ReadKey;
  If Reply in ['Д','д','L','l'] then
    IndPrinter:=true;

{ Уведення і друк вхідних даних }
  n:=0;
  While not eof(FileText) do
    Begin
      Inc(n);
```

```
      Readln(FileText,St[n]);
    End;
  ScreenText('Вхідний текст');
  If IndPrinter then
    PrinterText;

{ Установка таблиці результатів у початковий стан }
  For i:=1 to 26 do
    Begin
      TabNumberWord[i]:=0;
      TabWord[i]:='';
    End;

{ Формування таблиці результатів і вилучення відзначених слів }
  MakeTabResult;
  ScreenText('Текст після вилучення відзначених слів');

{ Запис таблиці результатів у вихідний файл }
  WriteTabResult;

{ Вилучення з тексту надлишкових пропусків }
  DeleteSpaces;
  ScreenText('Текст після вилучення надлишкових пропусків');

{ Вирівнювання скоректованого тексту по правій границі }
  RightBorder;
  ScreenText('Текст після вирівнювання по правій границі');

{ Запис перетвореного тексту у вихідний файл }
  S1:=' ';
  Write(FileOut,S1);
  S1:='         П Е Р Е Т В О Р Е Н И Й    Т Е К С Т';
  Write(FileOut,S1);
  For i:=1 to n do
    Write(FileOut,St[i]);

{ Читання і друк вихідного файлу }
  ScreenOutFile;
  If IndPrinter then
    PrinterOutFile;

{ Закриття файлів }
  Close(FileText); Close(FileOut);

End.
```

В х і д н и й т е к с т

ddfjh 5678 fjhiiii,AADCv .rty 8765 jhfb rty acd vbnnm qwe rrr vbnmnm qwer,ttyui op 1234 ajdfj hjkl rxcv vbnm pdfg rty w fghju 6789 890 hjk fder vbmnd nbnvbfdf iyutzz dfg n vcv cvvbi adfgh yuiop. TybGF JYbfbgn 9870 tty 12986 ghytn tRRew piuyt nbg nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb ijhghfgw kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg

Текст після вилучення відзначених слів

5678 , . 8765 rty acd
rrr vbnmnm qwer, 1234 ajdfj vbnm rty w
fghju 6789 890 hjk fder vbmnd nbnvbfdf dfg n vcv cvvbi
adfgh yuiop. TybGF JYbfbgn 9870 tty 12986 ghytn tRRew piuyt nbg
nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb
ijhghfgw kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg

Текст після вилучення надлишкових пропусків

5678,. 8765 rty acd
rrr vbnmnm qwer, 1234 ajdfj vbnm rty w
fghju 6789 890 hjk fder vbmnd nbnvbfdf dfg n vcv cvvbi
adfgh yuiop. TybGF JYbfbgn 9870 tty 12986 ghytn tRRew piuyt nbg
nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb
ijhghfgw kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg

Текст після вирівнювання по правій границі

5678,. 8765 rty acd rrr vbnmnm qwer, 1234 ajdfj vbnm rty w fghju 6
cvvbi adfgh yuiop. TybGF JYbfbgn 9870 tty 12986 ghytn tRRew piuyt
nbg nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb ijhghfgw
kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg

ТАБЛИЦЯ ВИХІДНИХ РЕЗУЛЬТАТІВ

N п/п	Буква	Номер слова	Слово
1	A	3	AADCv
2	B	5	jhfb
3	C	3	AADCv
4	D	1	ddfjh
5	E	9	qwe
6	F	1	ddfjh
7	G	19	pdfg
8	H	1	ddfjh
9	I	2	fjhiiii
10	J	1	ddfjh
11	K	16	hjkl
12	L	16	hjkl
13	M	8	vbnnm
14	N	8	vbnnm
15	O	14	op
16	P	14	op
17	Q	9	qwe
18	R	4	rty
19	S	0	
20	T	4	rty
21	U	13	ttyui
22	V	3	AADCv
23	W	9	qwe
24	X	17	rxcv
25	Y	4	rty
26	Z	27	iyutzz

П Е Р Е Т В О Р Е Н И Й Т Е К С Т

5678,. 8765 rty acd rrr vbnmnm qwer, 1234 ajdfj vbnm rty w fghju 6

```
cvvbi adfgh yuiop. TybGF JYbfbgn 9870 tty 12986 ghytn tRRew piuyt
nbg nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb ijhghfgw
kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg
```

Умова завдання 2. До складу вхідного тексту входять слова і цілі десяткові числа. Як роздільники в тексті використовуються пропуск, крапка, кома, крапка з комою, двокрапка, знак запитання і знак оклику. Кожне число має не більш ніж дев'ять десяткових цифр. Перед числом може стояти знак '+' або '-'. Потрібно перетворити числа в шістнадцятеричну систему числення і записати їх у вихідний файл, розділяючи між собою комами. Як ознаку шістнадцятеричного числа використати префікс $. При перетворенні чисел їхні знаки повинні бути збережені.

У складі програми Cont4_2 можна відзначити два принципових моменти: перетворення цілого 10-го числа в 16-у систему числення і вирівнювання рядків вихідного файлу по правій границі. Перший з них досить ясний з тексту процедури Make16, на другому потрібно зупинитися окремо.

У програмі Cont4-1 вирівнювання по правій границі виконується після закінчення формування рядків перетвореного тексту, тому що вилучення окремих елементів виконується безпосередньо в рядках вхідного тексту.

У програмі Cont4_2 масив рядків перетвореного тексту не формується, замість цього формуються поступово рядки вихідного файлу в міру перегляду вхідного тексту.

Черговий рядок, що надалі переноситься у виходний файл, - це рядок Sd. До початку обробки тексту йому привласнюється порожнє значення, а його довжина len приймається рівною нулю. Після одержання в рядку Sb чергового 16-го числа проводиться перевірка, чи можна рядок Sb дописати до рядка Sd (len+l<66). Якщо це підтверджується, то Sb дописується в Sd з додаванням коми для поділу з наступним елементом, при цьому збільшується відповідним чином довжина len. Якщо len+l>=66, то це означає, що чергове 16-е число Sb не може бути дописане в рядок Sd. Тоді цей рядок пересилається у вихідний файл FileOut, а рядку Sd привласнюється значення рядка Sb, тобто туди пересилається число, що не могло розміститися в попередньому рядку вихідного файлу.

Пересилка рядка Sd у вихідний файл відбувається лише після повного заповнення цього рядка, при цьому після такого пересилання в Sd записується чергове 16-е число. Отже, після закінчення циклу **While** рядок Sd не може бути порожній, у зв'язку із чим він додатково пересилається у вихідний файл. При цьому попередньо в рядку Sd вилучається останній символ, оскільки записувати кому після останнього числа не має сенсу.

```
Program Cont4_2;
Uses Crt,Printer;
Const
    Nmax= 50;       { макс. кількість ел-тів у масиві рядків St }
    Enter = 13;     { код клавіші Enter }
    Escape = 27;    { код клавіші Esc }
    PressKey = 'Натисніть клавішу Enter';
    Separs = ' .,;:?!'; { список роздільників }
Type
    String66 = string[66];    { тип рядка тексту }
    StringAr = array[1..Nmax] of String66;
Var
    i,                      { параметр циклу }
```

```
    n,                    { кількість рядків у масиві St }
    l,len,                { довжина рядка }
    k1,k2                 { границі слова }
          : byte;
    Code : integer;       { ознака перетворення }
    x : longint;          { число з тексту }
    Cond,                 { керуюча змінна }
    IndPrinter : boolean; { індикатор використання принтера }
    ch,                   { символ тексту }
    Reply : char;         { символ відповіді на запит програми }
    S,                    { рядок вхідного тексту }
    Sb,                   { чергове слово тексту }
    Sd : string66;        { рядок для запису у вихідний файл }
    St : StringAr;        { масив рядків тексту }
    FileText : text;      { вхідний файл }
    FileOut               { вихідний файл }
             : file of String66;
{ ------------------------------------------------------------ }
Procedure WaitEnter;
{ Затримка виконання програми, поки не буде натиснута }
{ клавіша Enter                                        }
Var ch : char;
Begin
  Repeat
    ch:=ReadKey;
  Until ord(ch) = Enter;
End { WaitEnter };
{ ------------------------------------------------------------ }
Procedure ScreenText;
{ Вивід на екран масиву рядків St }
Var  i : integer;
Begin
  ClrScr;
  Writeln('              В Х І Д Н И Й  Т Е К С Т');
  For i:=1 to n do
    Writeln(St[i]);
End { ScreenText };
{ ------------------------------------------------------------ }
Procedure PrinterText;
{ Друк на принтері вхідного тексту }
Var  i : integer;
Begin
  Writeln(Lst);
  Writeln(Lst,'              В Х І Д Н И Й  Т Е К С Т');
  For i:=1 to n do
    Writeln(Lst,St[i]);
End { PrinterText };
{ ------------------------------------------------------------ }
Procedure ScreenOutFile;
{ Вивід на екран вихідного файлу }
Var S : string66;
Begin
  Seek(FileOut,0);
```

```
  Writeln('              В И Х І Д Н И Й   Ф А Й Л');
  While not eof(FileOut) do
    Begin
      Read(FileOut,Sb);
      Writeln(Sb);
    End;
End { ScreenOutFile };
{ ------------------------------------------------------------ }
Procedure PrinterOutFile;
{ Друк на принтері вихідного файлу }
Begin
  Seek(FileOut,0);
  Writeln(Lst); Writeln(Lst);
  Writeln(Lst,'              В И Х І Д Н И Й   Ф А Й Л');
  While not eof(FileOut) do
    Begin
      Read(FileOut,S1);
      Writeln(Lst,S1);
    End;
End { PrinterOutFile };
{ ------------------------------------------------------------ }
Function SignBegin(S:string66; k:byte):byte;
{ Пошук початку слова в рядку S }
Var i : byte;
Begin
  SignBegin:=0;
  For i:=k to length(S) do
    If Pos(S[i],Separs)=0 then
      Begin
        SignBegin:=i; Exit
      End;
End { SignBegin };
{ ------------------------------------------------------------ }
Function SignEnd(S:string66; k:byte):byte;
{ Пошук кінця слова в рядку S }
Var i : byte;
Begin
  SignEnd:=0;
  For i:=k to length(S) do
    If Pos(S[i],Separs)>0 then
      Begin
        SignEnd:=i; Exit
      End;
End { SignEnd };
{ ------------------------------------------------------------ }
Function Make16(Sr:string66):string66;
{ Перетворення числа в 16-у с/ч }
Const  Dig16 : array[0..15] of char =
             ('0','1','2','3','4','5','6','7',
              '8','9','A','B','C','D','E','F');
Var    k : byte;
       d : char;
      Sm : string66;
```

```
Begin
  If (Sr[1]='+') or (Sr[1]='-') then   { перетворення рядка }
    Val(Copy(Sr,2,length(Sr)-1),x,Code){ Sr у десяткове число }
  Else                                 { x типу longint }
    Val(Sr,x,Code);
  Sm:='';
  Repeat
    k:=x mod 16;    { формування чергової 16-ої цифри k, }
                    { починаючи з молодшої цифри 16-го числа }
    x:=x div 16;
    d:=Dig16[k];    { перетворення цифри в символ і включення }
    Sm:=d+Sm;       { його до складу 16-го числа як рядка }
  Until x=0;
  Sm:='$'+Sm;       { додавання префікса}
  If (Sr[1]='+') or (Sr[1]='-') then  { додавання знака }
    Sm:=Sr[1]+Sm;
  Make16:=Sm;
End { Make16 };
{ ------------------------------------------------------------ }
Begin

{ Установлення відповідності між внутрішнім }
{ і зовнішнім файлами }
  ClrScr;
  Assign(FileText,'InFile.txt');
  Assign(FileOut,'OutFile.dat');

{ Відкриття використовуваних файлів }
  Reset(FileText); Rewrite(FileOut);

{ Запит про використання принтера }
  IndPrinter:=false;
  Writeln('Буде використатися принтер ?  (Так,Ні)');
  Reply:=ReadKey;
  If Reply in ['Д','д','L','l'] then
    IndPrinter:=true;

{ Уведення і друк вхідних даних }
  n:=0;
  While not eof(FileText) do
    Begin
      Inc(n);
      Readln(FileText,St[n]);
    End;
  ScreenText('Вхідний текст');
  If IndPrinter then
    PrinterText;

{ Пошук чисел у тексті іперетворення їх в 16-у с/ч }
  Sd:=''; len:=0;
  For i:=1 to n do
    Begin
      S:=St[i];
```

```
      Cond:=true; k2:=0;
      While Cond do
        Begin
          k1:=SignBegin(k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=SignEnd(k1+1);
              If k2=0 then
                Begin
                  k2:=length(S)+1; Cond:=false
                End;
              Sb:=Copy(S,k1,k2-k1);
              Val(Sb,x,Code);
              If Code=0 then          { це число, а не слово }
                Begin
                  Sb:=Make16(Sb);
                  l:=length(Sb);
                  If len+l<66 then    { формування рядка }
                    Begin             { вихідного файлу }
                      Sd:=Sd+Sb+','; Inc(len,l+1);
                    End
                  Else
                    Begin
                      If (i=n) and (not Cond) then
                        Delete(Sd,length(Sd),1);
                      Write(FileOut,Sd);
                      Sd:=Sb+','; len:=l+1;
                    End;
                End;
            End;
        End;
    End;
  Delete(Sd,length(Sd),1);    { вилучення останньої коми }
  Write(FileOut,Sd);

{ Читання і друк вихідного файлу }
  ScreenOutFile;
  If IndPrinter then
    PrinterOutFile;

{ Закриття файлів }
  Close(FileText); Close(FileOut);
  WaitEnter;

End.
```

В Х І Д Н И Й Т Е К С Т

```
abcdfr 8765  76,4567899; iutrew bnnn! 123456789 987654321
97531 13579, jhgfdsiuytrreew? 24680 8642 rtyuhbjj 111111
555 7777777: 666666. lkjhgfgdfsasasg ijhgff ytrtre 3333333
99999999 8888888,,,,44444444 33333 222222222 111111111 uiytr
```

```
765765765  123123123  234234234  987987987  qwertyujh 765765765

                    В И Х І Д Н И Й  Ф А Й Л
$223D,$4C,$45B35B,$75BCD15,$3ADE68B1,$17CFB,$350B,$6068,$21C2,
$1B207,$22B,$76ADF1,$A2C2A,$32DCD5,$5F5E0FF,$87A238,$2A62B1C,
$8235,$D3ED78E,$69F6BC7,$2DA4A885,$756B5B3,$DF6217A,$3AE38013
```

Варианти завдань

---

1. Для поділу слів у вхідному тексті українською мовою використаються пропуск, крапка, кома, крапка з комою, двокрапка і тире, при цьому кожний з роздільників може бути оточений ліворуч і (або) праворуч одним або декількома пропусками. Перенести у вихідний файл слова, які починаються й закінчуються голосною буквою, крім слів, які мають довжину менше трьох літер. Слова у вихідному файлі розділяти між собою комою, після якої записувати один пропуск.

---

2. Вхідний текст містить список цілих десяткових чисел, розділених комами. Перенести у вихідний файл паліндроми, тобто числа, які читаються однаково зліва направо і справа наліво (наприклад, 4554, 78987, 8 і т.п.). Якщо яке-небудь із вхідних чисел містить незначащі нулі, то такі нулі повинні бути попередньо вилучені. У вихідному файлі після поділяючої коми записувати один пропуск.

---

3. У кожному рядку вхідного тексту обміняти місцями перше й останнє слово. При цьому передбачається, що в даному тексті можуть бути переноси слів. Якщо останнє слово даного рядка містить перенос на наступний рядок, то обміну підлягають перше й передостаннє слова. Якщо перше слово даного рядка є продовженням останнього слова попереднього рядка, то обмінюються друге й останнє слова. Якщо перше й останнє слова даного рядка не є повними словами, то піддаються обміну друге й передостаннє слова. Вважати, що в кожному рядку є не менше двох повних слів.

Рекомендація. Для обміну слів використовувати буферний рядок.

---

4. Український текст містить кілька чисел, записаних у римській системі числення (діапазон чисел 1..3000). Перевести кожне число в десяткову систему числення.

Примітка. Як ознака римського числа можна прийняти те, що кожний символ у записі такого числа - це велика латинська буква (I, V, X, L, C, D, M). При визначенні числа застосовується таке правило: якщо дана римська цифра менша наступної, то її значення віднімається від числа, у противному випадку - додається. Наприклад, для числа MMCMXLVIII одержимо 1000+1000-100+1000-10+50+5+1+1+1 = 2948.

---

5. Вхідний текст містить два речення, кожне з яких закінчується крапкою. Речення може займати кілька рядків, причому в останньому рядку першого речення може починатися друге речення. Для запису слів використовуються малі латинські букви. Слова розділені між собою одним або декількома пропусками. Визначити набір символів, які повинні бути додані до першого речення, щоб з його складу можна було сконструювати друге речення.

---

6. У тексті для запису слів використовуються великі і малі латинські букви, які вважаються еквівалентними. До складу слів можуть входити також цифри. Слова розділені між собою одним або декількома пропусками. У кожному зі слів вхідного тексту згрупувати повторювані символи в тім порядку, у якому вони зустрічаються в даному слові. Наприклад, для слова

'Ax56KLdFa5XPUYK5XZy67DfUYPaJG765VRSZ'

одержимо

'AaaxXX5555666KKLdDFfPPUUYyYZZ77JGVRS'.

Якщо слово створене лише одним символом (наприклад, 'aaAAaaAa'), то таке слово вилучити із складу тексту.

---

7. Кожен рядок вхідного тексту - це безперервна послідовність цифр. Кожні дві цифри - це порядковий номер букви латинського алфавіту; код, що рівняється 00 або перевищує 26, вважається кодом пропуску. Виконати декодування вхідного тексту, замінивши кожне двозначне число відповідною буквою. Якщо який-небудь рядок містить непарну кількість цифр, то вставити перед останнім символом цифру '0'. Рядки вихідного файлу повинні бути вирівняні по правій границі.

---

8. Вхідний текст складається з українських букв, цифр і символів-роздільників (пропуск, крапка, кома і т.п.). Для запису слів використовуються великі і малі літери, що вважаються еквівалентними. Визначити в абсолютних і відносних одиницях частоту кожної букви і записати у вихідний файл у вигляді таблиці отримані результати в порядку зменшення відносної частоти букв. Як відносну одиницю виміру використати відсотки, при цьому записувати їх у таблицю із двома цифрами в дробовій частині. Слова, які містять у собі букву з найменшою, але не нульовою, відносною частотою, вилучити із складу тексту.

---

9. Вхідний текст містить список ідентифікаторів. Елементи списку розділені між собою комами, ліворуч і (або) праворуч від коми можуть бути один або кілька пропусків. У складі ідентифікатора можуть бути як великі, так і малі літери. Перенести в перетворений текст список помилкових ідентифікаторів, як роздільник між ними використати кому із пропуском.

Примітка. Ідентифікатор складається з латинських букв, цифр і знака підкреслення, при цьому починатися він повинен з букви. Знак підкреслення не може бути останнім, після цього знака повинна бути буква або цифра.

---

10. Вхідний текст містить слова, які складаються з малих латинських букв, цілі десяткові числа й роздільники (пропуск, кома, крапка і т.п.). До складу слова можуть входити цифри, але в цьому випадку першим символом повинна бути буква. Вилучити з тексту всі слова з непарними порядковими номерами, якщо довжина цих слів не перевищує 5 символів. У словах з парними порядковими номерами переставити букви у зворотному порядку, якщо ці слова не мають у своєму складі цифр. Нумерації підлягають лише слова, а не числа.

---

11. Слова вхідного тексту складаються із великих і малих латинських букв. У кожнім слові записати по алфавіту букви, які входять до його складу, вважаючи при цьому, що великі і малі літери еквівалентні. Наприклад, для слова

'AHGgfraUYTEuerfSTssASGgXYzxyZX'

одержимо

'AaAEeffGgGgHrrSssSTTUuXxXYYyz'.

Якщо в слові всі букви однакові, то вилучити його з тексту.

---

12. Слова в тексті розділені між собою одним з таких символів: пропуск, крапка, кома, крапка з комою і двокрапка, при цьому кожний із цих роздільників може бути оточений ліворуч і (або) праворуч одним або декількома пропусками. Визначити середню довжину *S* слів вхідного тексту і перенести в перетворений текст слова, довжина яких не перевищує значення *S*. Слова в новому тексті розділяти між собою комою з пропуском.

13. Вхідний текст містить слова українською мовою і цілі четверичні числа. Потрібно перетворити ці числа в десяткову систему числення, після чого переставити їх цифри у зворотному порядку. Перетворені числа не повинні містити незначащих нулів. До слів, які мають дві або три букви, дописати в їхній кінець ці ж букви, але у зворотному порядку. Отриманий новий текст повинен бути вирівняний по правій границі.

---

14. Вхідний текст містить слова, записані латинськими буквами. Для запису слів використовуються як великі, так і малі літери, які вважаються еквівалентними. Роздільниками в тексті є пропуск, крапка, кома, крапка з комою і двокрапка, при цьому кожний з них може бути оточений ліворуч і (або) праворуч одним або декількома пропусками. Перенести в перетворений текст слова вхідного тексту, які містять подвоєні голосні. Слова в цьому тексті розділяти комою з пропуском, кожне слово повинне починатися із великої букви, всі інші букви в слові повинні бути малими.

---

15. Текст містить слова й цілі десяткові числа, розділені між собою одним або декількома пропусками. Слова формуються з малих латинських букв. До складу слів можуть входити також цифри, але в цьому випадку першим символом є буква. Перед числом може стояти знак '+' або '-'. У вихідний файл записати спочатку слова, згруповані за алфавітом, а потім числа, згруповані по зменшенню кількості їхніх цифр. При цьому у вихідний файл переносити лише слова, які не мають у своєму составі цифр, і числа без знака. Рядки вихідного файлу повинні бути вирівняні по правій границі. Вважати, що як слова, так і числа не можуть мати більше ніж 15 символів.

Примітка. Рекомендується спочатку створити масив слів і масив чисел, згрупувати їх відповідно до умови завдання, а потім перенести у вихідний файл.

---

16. Вхідний текст містить список арифметичних констант у формі десяткових чисел із плаваючою комою. Елементи списку розділені між собою одним або декількома пропусками. Перед значенням порядку може бути записаний знак «+» або знак «-». Перед мантисою числа також може бути зазначено її знак. Перенести в перетворений текст список констант, які перевищують по модулю припустиме для типу real значення (1E38). Елементи списку у вихідному файлі розділити між собою комами, а після кожної коми записати один пропуск. Для аналізу констант у цілому процедури `Val` і `Str` не використовувати, для перетворення строкового значення порядку в число можна використати процедуру Val. Наприклад, для вхідного списку

1.2E+12-43.677E37 2567.234E+34 0.00087E45 14.6E-5 -6.6E+38 777E101

одержимо -43.677E37, 2567.234E+34, 0.00087E45, -6.6E+38, 777E101.

---

17. У тексті містяться слова українською мовою, розділені між собою пропуском, комою або крапкою. Для запису слів використовуються як малі, так і великі букви, що вважаються еквівалентними. Перенести у вихідний файл слова, які починаються із голосної букви, переставивши при цьому букви кожного слова у зворотному порядку. Слова у вихідному файлі повинні бути розділені між собою одним пропуском.

---

18. Слова вхідного тексту розділені між собою одним або декількома пропусками. Серед символів слова можуть бути як букви, так і цифри, але першої завжди записується буква. Аналізуючи текст у цілому, виконати перестановку його слів у зворотному порядку. Якщо серед символів слова є хоч одна цифра, вилучити таке слово із складу тексту. У перетвореному тексті слова повинні бути розділені між собою лише одним пропуском.

Вказівка. Спочатку виконати перестановку слів у кожному рядку, а потім переставити у зворотному порядку компоненти масиву рядків.

19. Вхідний текст містить список десяткових констант у формі чисел із плаваючою комою. Елементи списку розділені між собою одним або декількома пропусками. Записати ці константи у вихідному файлі у вигляді цілої й дробової частин, розділених крапкою. Якщо при цьому загальна кількість цифр у перетвореній константі більше 15, то її у вихідний файл не включати. Елементи вихідного файлу розділяти між собою комою, після коми записувати один пропуск. Наприклад, для вхідного списку

35.31E8 35.31E20 +35.31E-8 35.31E-20 123456789.987654321E+0-0.5E-3 1.48E+01

одержимо 3531000000.0, +0.0000003531, -0.0005, 14.8.

---

20. Вхідний текст містить кілька арифметичних виразів, до складу яких входять ідентифікатори простих змінних, цілі десяткові числа, знаки арифметичних операцій і круглі дужки. Вирази розділені між собою крапкою з комою. Між елементами виразів можуть бути пропуски.В одному рядку може бути кілька виразів, при цьому перенос частини виразу на наступний рядок не допускається. Проаналізувати в кожнім виразі баланс відкриваючих і закриваючих дужок і в випадку його порушення вилучити надлишкові зовнішні дужки. Вилучити також пропуск між елементами виразів. У перетвореному тексті після кожної крапки з комою записати один пропуск. Перетворений текст повинен бути вирівняний по правій границі, що спричиняє перенос частини виразу на наступний рядок.

---

21. Для запису слів у вхідному тексті використовуються великі і малі латинські букви, які вважаються еквівалентними. Роздільниками слів є пропуск, крапка, кома, крапка з комою, двокрапка і тирі, при цьому кожний із роздільників може бути оточений ліворуч і (або) праворуч одним або декількома пропусками. Перенести у вихідний файл із вхідного тексту пари слів, які складаються з однакових букв (наприклад, ABbxXy і xYxba). Між словами однієї пари ставити тире, пари слів відокремлювати одну від іншої комою з пропуском. Для запису слів у вихідному файлі використовувати лише малі літери. Приклад: abbxxy - xyxbab.

Примітка. Умова завдання означає, що в кожне з пари слів повинні входити не тільки однакові букви, але і їхня кількість також повинна бути однаковою. Наприклад, пара слів 'abbxxy' і 'abxxyy' не відповідає заданій умові.

---

22. Текст українською мовою містить список українських і російських прізвищ чоловічого роду. Елементи списку розділені між собою одним або декількома пропусками. Перетворити список таким чином, щоб спочатку йшли всі прізвища із закінченням -ко, потім - прізвища із закінченням -ін, у третій групі - прізвища із закінченням -ов, в останній групі - всі інші прізвища. У вихідний файл прізвища із закінченням -ук і -юк не пересилати. Відносна послідовність прізвищ у кожній групі повинна бути такою ж, як і у вхідному списку. Елементи нового списку розділяти між собою комою з пропуском.

Рекомендація. Створити 4 буферних файли, а потім об'єднати їх в один вихідний файл, знищивши зайві файли.

---

23. У тексті, записаному латинськими буквами, можуть бути групи слів, укладені в круглі дужки. В одному рядку може бути кілька груп таких дужок. Якщо в рядку знайдена відкриваюча дужка, але не виявлена закриваюча, то вважається, що закриваюча дужка неявно розташована після останнього символу рядка. Між словами тексту як роздільники можуть бути пропуск, кома або крапка, при цьому кожний з них може бути оточений одним або декількома пропусками. При формуванні перетвореного тексту вилучити з нього зазначені групи слів (разом з дужками).

---

24. Слова тексту розділені пропусками. Кожне слово – це послідовність алфавітно-цифрових символів, наприклад, A2B5E33426XRVT0Z11U. Цю вхідну послідовність потрібно інтерпретувати в такий спосіб: якщо черговий символ – цифра *n* (*n* = 0..9), то це розглядається як вказівка про *n* повторень наступного символу незалежно від того, чи є цей символ цифрою або довільною буквою; при цьому повторюваний символ більше не розглядається як можливий коефіцієнт повторення. Для наведеного вище приклада маємо

ABBEEEEE3332222XXXXXXRVT1U

Потрібно всі слова тексту перетворити за описаним вище правилом і записати їх у вихідний файл. Слова у вихідному файлі повинні розділятися між собою комою з пропуском.

---

25. У вхідному тексті безперервним потоком записані групи латинських букв і групи цифр. При цьому застосовуються як великі, так і малі літери, які вважаються еквівалентними. Відокремити кожну групу одним пропуском від сусідніх груп. Визначити кількість груп цифр, кількість груп букв, а також кількість різних букв і цифр, які використані в тексті.

---

26. Вхідний текст містить слова українською мовою й цілі десяткові числа, розділені між собою одним або декількома пропусками. Для запису слів використовуються як малії, так і великі букви. У кожному числі переставити його цифри у зворотному порядку. Вилучити із числа незначащі нулі, якщо вони утворилися після перестановки цифр. У словах тексту замінити великі букви малими, за винятком першої букви слова, якщо вона велика. Як роздільник у перетвореному тексті використовувати лише один пропуск.

---

27. У вхідному тексті використовуються тільки латинські букви (великі і малі) і роздільники (пропуск, кома, крапка, крапка з комою). Згрупувати слова цього тексту в порядку збільшення відносної кількості в них голосних букв. Вважати, що довжина окремого слова не перевищує 20 символів. Слова в новому тексті розділяти між собою комою з пропуском. Як одиницю відносної кількості використати відсотки, дробова частина яких має дві цифри.

---

28. Для поділу слів у вхідному тексті використовуються пропуск, крапка, кома, крапка з комою, двокрапка і тире, при цьому кожний із роздільників може бути оточений ліворуч і (або) праворуч одним або декількома пропусками. Перенести у вихідний файл із вхідного тексту слова, у яких всі букви різні. Якщо слово має хоча б одну цифру, то воно переносу не підлягає. У перетвореному тексті розділяти слова лише одним пропуском.

---

29. Вхідний текст містить список шістнадцятеричних чисел, розділених між собою комами. Ціла й дробова частина кожного числа розділені крапкою. Переставити цифри числа у зворотному порядку, окремо для цілої й дробової частин. Вилучити незначащі нулі, якщо вони з'явилися при перестановці цифр. Якщо в дробовій частині числа немає значущих цифр, то вилучити із складу числа також поділяючу крапку. Після виконання зазначених дій перетворити числа, що містяться в тексті, у десяткову систему числення, при цьому дробова частина числа повинна мати не більше ніж 4 цифри.

---

30. У вхідному тексті задано список десяткових констант у формі цілої й дробової частин, розділених крапкою. Елементи списку розділені між собою одним або декількома пропусками. Записати ці константи у формі з плаваючою комою, передбачивши в цілій частині мантиси одну значущу цифру (не нуль). У записі порядку числа обов'язково вказувати після букви 'E' знак порядку '+' або '-'. Значення порядку записувати з двома цифрами.

Примітка. Виключенням є число 0, воно зображується у вигляді 0.0E+00.

---

31. Вхідний текст містить слова, які складаються із великих і малих латинських букв, числа, знаки арифметичних операцій ('+', '-', '*', '/'), розділові знаки (кома, крапка та ін.) і дужки. Сформувати новий текст, записавши в нього суцільним потоком три групи символів (букви, цифри, інші символи), дотримуючись при цьому їх вхідної послідовності. Пропуски в новий текст не переносити. Кожна зі згаданих вище груп символів може займати більше одного рядка. Групи розділити між собою крапкою з комою, після якої записати один пропуск.

Рекомендація. Тричі переглядати вхідний текст, переносячи при цьому у вихідний файл відповідні символи.

---

32. Слова тексту формуються великими і малими латинськими буквами, як роздільники слів використовуються пропуск, кома і крапка, при цьому кожний із них може бути оточений одним або декількома пропусками. Букви, з яких складаються слова вхідного тексту, згрупувати в нові слова по *k* літер у кожному (змінну *k* увести із клавіатури, передбачивши її значення в діапазоні 1..10). При формуванні нових слів необхідно зберегти вхідну відносну послідовність букв. Кожен рядок вхідного тексту вважається продовженням попереднього рядка. Слова нового тексту розділяти між собою комою з пропуском.

Наприклад, для вхідного тексту

Fgrty, uytkjhg treLKhfdui, nhSDgjkll.hjgkj.jhkliom,hjkkll

при $k=3$ одержимо

Fgr, tyu, ytk, jhg, tre, LKh, fdu, inh, SDg, jkl, lhj, gkj, jhk, lio, mhj, kkl, l

Рекомендація. Вилучити із вхідного тексту всі роздільники, після чого групувати його символи відповідно до умови завдання.

---

33. У вхідному тексті використовуються великі і малі українські букви, які вважаються еквівалентними. У складі слів можуть перебувати цифри. Роздільниками слів є пропуск, кома, крапка, крапка з комою і двокрапка, які можуть бути оточені ліворуч і (або) праворуч одним або декількома пропусками. Слова, що не містять у своєму складі цифри, необхідно перенести в перетворений текст, якщо кількість приголосних у слові більше або дорівнює кількості голосних. У кожному із перенесених слів переставити букви у зворотному порядку.

---

34. У тексті безперервним потоком розташовані цифри та інші символи, при цьому кожен рядок не вважається продовженням попереднього рядка. Перенести у вихідний файл всі групи цифр вхідного тексту, розташувавши їх у порядку зменшення довжини групи. Вважати, що довжина безперервної групи цифр не перевищує 20 символів. Елементи перетвореного тексту розділяти між собою комою з пропуском.

---

35. Як роздільники в словах вхідного тексту використовуються пропуск, кома, крапка і крапка з комою, при цьому кожний із них може бути оточений одним або декількома пропусками. Перенести у вихідний файл слова, які складаються з тих же букв, що й перше слово тексту (не зважаючи при цьому на кількість повторень кожної букви). Перше слово вхідного тексту також повинне бути записане у вихідний файл. Слова в перетвореному тексті розділяти комою з пропуском.

---

36. Вхідний текст містить слова, які складаються з малих і великих латинських букв, цілі десяткові числа й роздільники (пропуск, кома, крапка з комою, двокрапка). Вилучити з кожного рядка тексту слова, які вже раніше зустрічалися в цьому рядку. При цьому великі й малі літери вважаються еквівалентними. У словах, що залишилися, перша

буква повинна бути великою, інші - малими. Слова в перетвореному тексті розділяти між собою лише одним пропуском.

---

37. Вхідний текст містить список цілих десяткових чисел, розділених між собою комами. Вилучити з тексту числа, які входять у діапазон $[m,n]$, $m > n > 0$, а інші числа перетворити у двійкову систему числення. Значення цілочисельних змінних m і n увести із клавіатури.

---

38. До складу вхідного тексту входять українські слова й цілі десяткові числа без знака. Вилучити з тексту ті слова, які починаються й закінчуються приголосною буквою, а кожне число замінити його цифровим коренем. При цьому враховувати, що великі й малі літери в даному тексті еквівалентні.

Примітка. Цифровим коренем називають таку суму цифр цілого числа, що після деякої кількості додавань зображується однією цифрою. Приклад:

$$853977 \to 8+5+3+9+7+7=39 \to 3+9=12 \to 1+2=3$$

---

39. Вхідний текст містить список десяткових констант із фіксованою крапкою. Константи розділені між собою комами. Перенести у вихідний файл список неправильних констант, розділивши їх на наступні групи по типу помилки:

- буква в першій позиції;
- використання неприпустимих символів;
- наявність в одній константі декількох поділяючих крапок;
- наявність більше одного знака '+' або '-';
- положення знака '+' або '-' у середині або наприкінці константи.

Кожна група неправильних констант повинна починатися з нового рядка вихідного файлу.

Примітка 1. Рекомендується 5 разів виконати перегляд вхідного файлу, пересилаючи по черзі у вихідний файл константи однієї групи.

Примітка 2. Константа може містити в собі кілька помилок. Тоді та сама константа буде записана в кілька груп.

---

40. Слова вхідного тексту записані великими і малими латинськими буквами, які вважаються еквівалентними. Слова розділені між собою одним або декількома пропусками. Перенести у вихідний файл слова, які при перестановці букв можуть стати паліндромами, розділяючи їх між собою комою з пропуском.

Примітка. Слово може бути паліндромом, якщо

- кількість кожної букви в слові парне;
- серед букв слова є лише одна, кількість якої непарне.

---

41. Для запису вхідного тексту використовуються тільки великі і малі латинські букви. Роздільниками між словами є пропуск, крапка і кома, які можуть бути оточені ліворуч і (або) праворуч одним або декількома пропусками. Знаки переносу в тексті не використовуються. Потрібно вирівняти текст по правій границі, вважаючи в цій якості позицію 66. Наявність абзаців у тексті не враховувати. Для вирівнювання тексту додати пропуски між словами, але таким чином, щоб їхня кількість скрізь була однаковою або відрізнялася не більше ніж на одиницю. Після останнього слова в рядку не повинно бути пропусків. Якщо між словами записані кома або крапка, то пропуск може бути тільки після такого символу.

Рекомендація. Спочатку залишити між словами в рядку лише по одному пропуску, після чого додати потрібну кількість пропусків.

---

42. Вхідний текст містить слова, а також десяткові й двійкові числа. Ознакою двійкового числа є буква 'B', записана після його цифр. Замінити двійкові числа на суму їхніх цифр, записавши цю суму також у двійковій системі числення. Вилучити з тексту десяткові числа, які мають менше трьох цифр. Новий текст переслати у вихідний файл, рядки якого повинні бути вирівняні по правій границі.

---

43. Вхідний текст містить список арифметичних констант у формі цілих десяткових чисел, розділених між собою комами. Зменшити значення кожної константи в 10 разів і записати її у вихідний файл. Врахувати такі окремі випадки:

- наприкінці константи є нулі;
- константа не кратна 10 і має більше однієї цифри;
- константа має лише одну цифру, але не дорівнює нулю;
- константа дорівнює нулю.

Наприклад, для списку '550, 234, 7, 0' одержимо '55, 23.4, 0.7, 0'.

Вказівка. Для перетворення констант не використовувати процедури `Val` і `Str`.

---

44. Вхідний текст містить кілька пар відкриваючих і закриваючих дужок. Перенести в перетворений текст фрагменти вхідного тексту, які містяться у всіх внутрішніх дужках. Якщо в цих фрагментах є пропуски, то їх необхідно вилучити. Фрагменти розділяти між собою крапкою з комою і пропуском. Наприклад, для вхідного тексту 'AB* (3+5/C8* (4-16*XYZ)+1)*(3-51*S*(6-8 * K18))' одержимо 4-16*XYZ; 6-8*K18. Вважати, що взятий із внутрішніх дужок текст не має продовження на наступному рядку рядку.

---

45. Вхідний текст містить слова українською мовою та цілі десяткові і вісімкові числа, розділені між собою одним або декількома пропусками. Ознакою вісімкового числа є буква 'X', записана після останньої цифри числа. Вилучити з тексту десяткові числа, які мають менше трьох цифр. Вісімкові числа перетворити в десяткову систему числення, після чого цифри отриманого числа переставити у зворотному порядку. Перетворені числа не повинні містити незначащих нулів.

---

46. Вхідний текст містить слова, які складаються з малих латинських букв, цілі десяткові числа й роздільники (пропуск, кома, крапка, крапка з комою). Вважаючи, що довжина окремого слова не перевищує 16 символів, сформувати таблицю слів вхідного тексту, указавши в ній кількість повторень кожного слова. Таблицю згрупувати за алфавітом слів. Після цього для повторюваних слів залишити в тексті лише їхній перший екземпляр, а інші вилучити з тексту. Перегляд вхідного масиву повинен виконуватися лише два рази: перший раз для формування таблиці слів, другий раз - для вилучення повторюваних слів.

---

47. Вхідний текст містить список десяткових чисел, у який входять цілі числа і числа з фіксованою крапкою (ціла й дробова частини числа розділені крапкою). Елементи списку розділяються комами. Деякі числа можуть містити знаки '+' або '-'. Визначити кількість додатних чисел, кількість від’ємних чисел, а також частоту кожної цифри в цілих частинах заданого списку чисел (у відсотках, у дробовій частині яких дві цифри). Після цього ті числа, які мають у складі своєї цілої частини цифру з найменшою, але не нульовою, частотою повторень, вилучити з тексту.

Примітка. Цілі десяткові числа, у тому числі нульові, не повинні впливати на визначення частоти цифр.

---

48. Вхідний текст містить список десяткових констант із фіксованою крапкою, при цьому кожна константа має явно виражені цілу й дробову частини. Елементи списку

розділені між собою комами. Необхідно в кожній константі вилучити незначащі нулі в цілій і дробовій частинах, якщо в її складі є такі нулі. Якщо дробова частина константи нульова, то вилучити також поділяючу крапку. Врахувати, що в цілій частині константи завжди повинна бути хоча б одна цифра. У вихідному файлі після будь-якої розділяючої коми записати один пропуск. Наприклад, для списку констант '00035.4500,0000.40,72.000,0.0' одержимо '35.45, 0.4, 72, 0'.

---

49. Слова тексту містять лише великі і малі латинські букви. Ці слова розділені між собою одним або декількома пропусками. Перетворити кожне слово таким чином, щоб у кожній парі букв (1-2, 3-4, 5-6 і т.д.) даного слова вони були розташовані за алфавітом. Якщо слово має непарну кількість букв, то останню букву не аналізувати. Наприклад, для слова 'diScreTioNary' одержимо 'dicSeriTNoary'. Якщо перша й остання букви перетвореного слова - голосні, то вилучити таке слово з тексту.

---

50. При записі тексту використовуються великі і малі латинські букви. Роздільниками слів є пропуск, кома, крапка, крапка з комою і двокрапка, які можуть бути оточені ліворуч і (або) праворуч одним або декількома пропусками. Визначити середню довжину S слова, а також середнє квадратичне відхилення G цієї довжини. Після цього вилучити з тексту слова, довжина яких $l > S + G$.

Примітка 1. Стосовно одномірного масиву

$$S = \frac{1}{n}\sum_{i=1}^{n} x_i \; ; \qquad G = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(S - x_i)^2}$$

Примітка 2. Для рішення поставленого завдання потрібно тричі переглядати слова вихідного тексту:

1) визначення параметра *S;*
2) визначення параметра *G;*
3) перенос у вихідний файл слів, для яких $l \le S + G$.
