МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

_

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО КОНТРОЛЬНОЇ РОБОТИ ПО КУРСУ ПРОГРАМУВАННЯ

ЧАСТИНА 3

(для студентів спеціальності 7.091501
«Комп'ютерні системи та мережі»
заочної форми навчання)

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

Затверджено
на засіданні навчально-
видавницької ради ДонНТУ
Протокол № 5 від 06.12.10

Донецьк ДонНТУ 2010

УДК 681.3(07)

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО КОНТРОЛЬНОЇ РОБОТИ ПО КУРСУ ПРОГРАМУВАННЯ, ЧАСТИНА 3 (для студентів спеціальності 7.091501 «Комп'ютерні системи та мережі» заочної форми навчання). Складачі: В.І.Назаренко, О.Ю.Чередніков, К.Б.Юсупова. – Донецьк, ДонНТУ, 2010. – 42 с.

Приведені методічні вказівки до контрольної роботи № 3 по курсу програмування для студентів-заочників спеціальності 7.091501. Контрольна робота містить завдання по темі «Програмна обробка текстів». Для виконання контрольної роботи наведено 50 варіантів завдань. Матеріал посібника містить також необхідний теоретичний матеріал і приклад виконання одного із завдань.

Складачі: доц.Назаренко В.І.
ас.Чередніков О.Ю.
ас. Юсупова К.Б.

Відповідальний
за випуск проф.Святний В.А.

Рецензент доц.Федяєв О.І.

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

Контрольна робота № 3 для студентів-заочників спеціальності 7.091501 "Комп'ютерні системи та мережі" містить у собі завдання, орієнтовані на розробку й виконання програм мовою Турбо Паскаль.

Контрольна робота, представлена студентом на кафедру, повинна містити:

- умову завдання;
- короткий опис алгоритму і програми;
- лістинг програми;
- результати рішення завдання.

У методичному посібнику наведено 50 варіантів завдань. Вибір варіанта виконується по останнім двом цифрам номера залікової книжки, збільшеним на 1. Якщо отримане число перевищує значення 50, то воно зменшується на 50.

Тексти контрольної роботи, у тому числі лістинги програм, треба виконувати на стандартних аркушах паперу формату А4. При цьому на титульному аркуші повинно бути написане:

Донецький національний технічний університет
Кафедра комп’ютерної інженерії

Контрольна робота № 3
за курсом "Програмування"

ст-та гр. (шифр групи)
Прізвище й ініціали

Номер залікової книжки ...

Домашня адреса: .

Текст програми (лістинг) і результати рішення завдання повинні бути видрукувані на принтері. Для цього доцільно використовувати безпосередньо можливості компілятора мови Турбо Паскаль. Якщо ж за якимось причинами друк тексту програми здійснюється за допомогою текстового процесора Word, то для цього потрібно використовувати шрифт Courier New розміром 12 (у шрифті Courier New кожний символ тексту займає окрему позицію). У тексті програми в обов'язковому порядку повинні бути виконані вимоги по стилю програмування (ці вимоги викладені в першій частині методичних вказівок).

Умова завдання та інші матеріали, що включаються у пояснювальну записку по контрольній роботі, рекомендується друкувати шрифтом Times New Roman розміром 14. При цьому для формул і інших математичних позначень доцільно використовувати редактор формул, що входить до складу текстового процесора Word.

Тексти методичних вказівок містяться не тільки у виданих типографським способом посібниках, вони записані також на комп'ютерах дисплейного класу. Студентам рекомендується переписати тексти методвказівок на дискету і використовувати програмні приклади, що містяться в них, як основу для розробки свого варіанта виконання завдання. У першу чергу ця рекомендація відноситься до сервісних процедур і функцій. Включення їх у свою програму шляхом копіювання файлу принаймні заощаджує час, нераціонально затрачуваний при наборі цих процедур із клавіатури.

М Е Т О Д И Ч Н І В К А З І В К И
по темі контрольної роботи

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

У всіх завданнях, якщо це особливо не застережено, передбачається обробка сторінки тексту, кількість рядків якої не перевищує 50, а кількість позицій у рядку не перевищує заданого значення (наприклад, 65). 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 передбачається (використання переносу слів помітно ускладнює програму обробки тексту).

Вхідний текст передбачається записаним у зовнішньому файлі, після уведення з файлу він представляється у вигляді масиву рядків. Перетворений відповідно до умови завдання текст повинен бути розміщений у тім же масиві рядків, що і вхідний текст (або в додатковому масиві рядків, якщо це потрібно по алгоритму рішення завдання), після чого він записується у вихідний файл. Якщо за умовою завдання потрібно формувати список яких-небудь елементів вхідного тексту (слів, констант і т.п.), то елементи списку у вихідному файлі, як правило, повинні бути розділені комами.

Якщо в процесі перетворення вхідного тексту змінюються довжини його рядків, то перед записом тексту у вихідний файл повинно бути виконане вирівнювання рядків тексту шляхом переносу у вільну праву частину даного рядка деякої початкової ділянки наступного рядка. Оскільки в завданнях до контрольної роботи №3 перенос слів не використовується, то в кожному рядку перетвореного тексту повинно бути розміщена максимально припустима ціла кількість слів. Зміна пропусків між словами для повного вирівнювання рядків тексту не виконується, якщо це спеціально не застережено в завданні.

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

У кожному завданні вхідний файл повинен бути описаний як текстовий, вихідний файл - як типізований, компонентами якого є рядки заданої довжини.

Операції, процедури і функції для обробки рядків

Стосовно рядків може бути використана лише одна операція - операція зчеплення (конкатенації), що позначається символом "+". Якщо змінні *S*1 і *S*2 оголошені як рядки, то вираз *S*1 + *S*2 означає, що до кінця рядка *S*1 дописується рядок *S*2. При цьому автоматично формується відповідне значення довжини об'єднаного рядка, записуване в його нульовому байті.

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);
  Writeln('     S4=',S4);
```

Буде видрукувано:

```
S3=0123456789abcde      { поточна довжина S3  15 символів }
S4=0123456789abcdefgh   { поточна довжина S4  18 символів }
```

Припустимо, що рядок *S*4 потрібно доповнити символами "*" (або пропусками) 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
    S4:=S4+'*';
  Writeln('     S4=',S4);
```

Буде видрукувано:

```
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 видрукувано:

```
     S4=0123456789abcdefgh
```

У прикладі 3 байти 21..25 рядка справді будуть заповнені символом "*", але поточна довжина рядка, на відміну від приклада 2, при цьому не змінюється (тут у циклі обробляються окремі символи рядка, а не рядок у цілому). Щоб результат обробки в прикладі 3 був таким же, як і в прикладі 2, потрібно після оператора **For** записати оператор *S*4[0]:=chr(25) (нульовий байт рядка, як і інші його байти, сприймається в програмі як символ; вищенаведений оператор пропонує занести в нульовий байт рядка *S*4 символ, що відповідає порядковому номеру 25 таблиці ASCII).

Для рядків припустимі всі операції відношення. Порівняння рядків виконується зліва направо по одному байту, поки не будуть виявлені різні байти. Нехай у цих байтах розташовані символи *C*1 і *C*2. Тоді

$$C1 < C2, \text{ якщо } \mathrm{ord}(C1) < \mathrm{ord}(C2).$$

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

Приклад 4.

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

Тут маємо *b*1 = *true*, b2 = *false*, *b*3 = *false*.

Малі і великі букви в рядку 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 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*: початкова 10; кінцева 7.

Процедура **Insert(S1,S2,Start)** - це вставка рядка *S*1 у рядок *S*2, 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);
  Writeln('   S1=',S1); Writeln('   S2=',S2);
  Writeln('   S3=',S3);
```

Буде видрукувано:

```
S1=xyz
S2=01xyz234
S3=01xyz23456
```

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

Примітка. Частковим випадком виразу є 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);
  Writeln('   S1=',S1); Writeln('   S2=',S2);
  Writeln('   S3=',S3); Writeln('   4=',S4);
```

Буде видрукувано:

```
S1=  15.68
S2= 7.000E+00
S3= 123456789. { дробова частина числа z не помістилася в S3 }
S4=     -789
```

Для зіставлення із процедурою Str розглянемо роботу процедури Write при виводі в текстовий файл (екран дисплея, 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 чисельної змінної з її внутрішньомашинного подання (відповідно до опису змінної в розділі **Var**) у рядок;
- вивід 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 першого неправильного символу, а змінній *X* при цьому буде привласнене нульове значення.

Приклад 8.

```
Var
  S1,S2,S3,S4,S5,S6 : string[10];
  x1,x2,x3 : real;
  k1,k2,k3,
  Code1, Code2, Code3,
  Code4, Code5, Code6 : integer;
Begin
  S1:='  15.48'; S2:='  15,48'; S3:='  15.ab';
  S4:='   678'; S5:='   6-78'; S6:='   6 78';
  Val(S1,x1,Code1); Val(S2,x2,Code2);
  Val(S3,x3,Code3);
  Val(S4,k1,Code4); Val(S5,k2,Code5);
  Val(S6,k3,Code6);
  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);
  Writeln('k3,Code6 = ',k3,'  ',Code6);
```

Буде видрукувано:

```
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  5
k3,Code6 = 0  5
```

Для зіставлення із процедурою Val розглянемо роботу процедури Read при уведенні з текстового файлу (клавіатура, диск). Число в текстовому файлі

представлено у вигляді 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;**
- пересилання значення змінної у відповідне поле пам'яті.

Перший етап роботи процедури Read еквівалентний роботі процедури Val.

Функція **Copy(S,Start,n)** - 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)** - це конкатенація (зчеплення) рядків *S*1, *S*2, ... , *Sn.* Дія функції Concat еквівалентна послідовному виконанню операцій зчеплення *S*1 + *S*2 +...+ *Sn.*

Функція **Length(S)** - це визначення поточної довжини рядка *S*. Дія цієї функції еквівалентна обчисленню виразу ord(S[0]) або byte(S[0]).

Функція **Pos(S1,S2)** визначає позицію, з якої відзначається перша поява в рядку *S*2 підрядка *S*1. Якщо *S*1 не міститься в *S*2, то вихідне значення 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);
  Writeln('k1,k2,k3 = ',k1,'  ',k2,'  ',k3);
```

Буде видрукуване

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

Функція UpCase(ch) виконує перетворення малої латинської букви в велику букву. 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]);
  Writeln('S = ',S);
```

Одержимо :

```
S = ABC0123KLMN+фбщг-UVWXYZ
```

Для ілюстрації використання перерахованих процедур і функцій розглянемо кілька прикладів.

У прикладах, що нижче приводяться, нерідко застосовуються процедури Inc (інкремент) і Dec (декремент).

Inc(i) еквівалентно i:=i+1;
Inc(i,n) еквівалентно i:=i+n;
Dec(i) еквівалентно i:=i-1;
Dec(i,n) еквівалентно i:=i-n.

Тут *i* - цілочисельна змінна, *n* - цілочисельний вираз, в окремому випадку - константа.

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

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.

Спочатку вирішимо аналогічне завдання для одновимірного цілочисельного масиву, у якому потрібно вилучити від’ємні елементи.

У програмі будемо послідовно переглядати елементи масиву *X* і, якщо черговий елемент *x*[*i*] < 0, то будемо робити зрушення підмасиву x[*i*+1]..*x*[*n*] на один елемент вліво, одночасно зменшуючи кількість елементів *n*.

```
Program Mas;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  i,j,n : integer;
          X : Ar;
Begin  Уведення n,X
  For i:=1 to n do
    If  x[i] < 0  then
      Begin
        For j:=i to n-1 do
          x[j]:=x[j+1];
        Dec(n);
      End;
```

```
End.
```

Із приводу програми Mas можна зробити три зауваження.

1. При $i = n$, коли аналізується останній елемент масиву, параметр j змінюється від n до n-1, тобто початкове значення параметру циклу менше його кінцевого значення. Як відомо з опису оператора **For**, цикл у цьому випадку не виконується, тобто оператор циклу еквівалентний порожньому операторові. Отже, при $i = n$ відбувається лише зменшення значення n, що відповідає алгоритму рішення завдання.

2. Початкове й кінцеве значення параметра циклу обчислюються до початку циклу і в процесі його виконання не змінюються. Отже, зменшення значення змінної n при $x[i] < 0$ не змінює кількість повторень циклу по параметру i. Це приводить до надлишкової роботи програми, але не відбивається на правильності рішення завдання.

3. Припустимо, що в масиві X є від'ємні елементи, що *підряд* ідуть, x[k] і x[k+1]. При $i = k$ буде вилучений елемент $x[k]$, а на його місце буде переміщено елемент $x[k+1]$, значення якого також менше нуля. Оскільки при новому повторенні циклу **For** параметр циклу приймає чергове значення k+1, те від'ємний елемент $x[k]$ не буде аналізуватися повторно і, як наслідок, не буде вилучений з масиву. Тому при наявності в масиві підряд розташованих від'ємних елементів програма Mas буде працювати неправильно.

Для правильного рішення поставленого завдання потрібно, щоб програма після вилучення від'ємного елемента $x[k]$ повторно аналізувала цей елемент і переходила до розгляду елемента $x[k+1]$ лише при $x[k]>=0$. Для цього потрібно в програмі замість циклу **For** використовувати цикл **While**:

```
Program Mas1;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  i,j,n : integer;
          X : Ar;
Begin  Уведення n,X
  i:=1;
  While  i <= n  do
    If  x[i] < 0  then
      Begin
        For j:=i to n-1 do
          x[j]:=x[j+1];
        Dec(n);
      End
    Else
      Inc(i);
End.
```

Правильна обробка масиву X буде зроблена також, якщо в циклі **For** виконувати перегляд масиву не зліва направо, а справа наліво (програма Mas2):

```
Program Mas2;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  i,j,n : integer;
         X : Ar;
Begin  Уведення n,X
  For i:=n downto 1 do
    If  x[i] < 0  then
      Begin
        For j:=i to n-1 do
          x[j]:=x[j+1];
        Dec(n);
      End;
End.
```

Вилучення з рядка пропусків аналогічне вилученню від'ємних елементів з одновимірного масиву. При цьому зрушення символів рядка вліво і зменшення його поточної довжини виконує процедура Delete.

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

Приклад 13. У рядку задано текст. Слова тексту розділені між собою одним або декількома 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 "-" використано тут як пропуск):

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

Ознакою початку слова є символ, відмінний від пропуску (непропуск), ознакою закінчення слова - найближчий до нього пропуск.

Будемо привласнювати номера позицій, що визначають положення слова в рядку, змінним *k*1 і *k*2:

```
-i-abcde----fghijklmn----n-opq--xy-i
   ^    ^   ^         ^
   k1   k2  k1        k2 ...
```

Отже, при аналізі рядка в програмі буде потрібно багаторазово визначати місця розташування слів або, інакше кажучи, позицію найближчого пропуску або непропуску, 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 пропуску або непропуску.

Пошук найближчого пропуску:

```
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 непропуску:

```
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 Slovo, що реалізує рішення поставленого завдання, містяться функції Space і NotSpace.

```
Program Slovo;
Uses Crt;
Var
   S : string;     { оброблюваний рядок }
   NumWords,       { кількість слів }
   NumLetters,     { кількість букв }
```

```
    k1,k2,          { ліва й права границі слова }
    l               { довжина слова }
        : byte;
    MiddleLet { середня кількість букв у слові }
        : real;
    Cond { умова закінчення обробки тексту }
        : boolean;
{ ---------------------------------------------- }
Function Space ............
Function 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;
          l:=k2-k1;
          Inc(NumWords);
          Inc(NumLetters,l);
        End;
    End;
  If NumWords>0 then
    MiddleLet:=NumLetters/NumWords;
  Writeln(' Кількість слів = ',NumWords,
          ' Кількість букв = ',NumLetters);
  Writeln('Середня довжина слова = ',MiddleLet:6:1);
End.
```

У програмі виконується послідовне виділення слів тексту шляхом визначення границь слова *k*1 і *k*2, де *k*1 - перша буква слова (перший непропуск), *k*2 - перший пропуск після кінця слова.

Стартове значення *k*2 приймається рівним нулю. Цикл пошуку слів триває доти, поки є істинним значення змінної *Cond*.

У кожному циклі пошуку визначаються значення *k*1 і *k*2. Значення *k*1 - це позиція найближчого непропуску, починаючи з байта *k*2+1 (у першому циклі *k*2+1 = 1). Якщо при звертанні до функції NotSpace отримано *k*1=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 *k*1>0, визначається значення *k*2, тобто номер позиції найближчого пропуску, починаючи з байта *k*1+1. Якщо при звертанні до функції Space отримано *k*2=0, то це означає, що до кінця рядка немає більше пропусків. Тоді правою границею слова є байт length(*S*)+1. Одночасно значення *k*2=0 указує, що в тексті знайдено останнє слово. Тоді змінній Cond привласнюється значення false, що веде надалі до припинення циклу пошуку.

В окремому випадку рядок *S* може бути порожній або містити одні пропуски. Тоді NumLetters = 0, NumWords = 0. Щоб запобігти перериванню програми через ділення на нуль, при обчисленні середньої кількості букв MiddleLet попередньо перевіряється значення змінної NumWords.

Приклад 14. У рядку містяться слова, що 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;
- суму десяткових цифр, що містяться в цих числах.

Методика пошуку початку числа в тексті аналогічна методиці пошуку початку слова в попередньому прикладі, але ознакою початку числа є цифра або символи "+","-". Тому замість функції NotSpace для пошуку початку числа будемо використовувати функцію Number:

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

Тут пошук по рядку *S* ведеться доти, поки не буде знайдений символ, що входить до складу рядка-константи *Sig*, що й визначає початок чергового числа.

```
Program Numbers;
Uses Crt;
Var
   S : string;      { оброблюваний рядок }
   i,               { параметр циклу }
```

```
    NumNumbers,      { кількість чисел }
    k1,k2,           { ліва й права границі числа }
    l,               { довжина числа }
    Dig              { числове значення цифри }
         : byte;
    Code : integer; { код перетворення }
    Sum : longint;  { сума цифр }
    Cond     { умова закінчення обробки тексту }
         : boolean;
{ ------------------------------------------- }
Function Space ............
Function 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+1,0);
          If k2=0 then
            Begin
              k2:=length(S)+1; Cond:=false;
            End;
          l:=k2-k1;
          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;
          Inc(NumNumbers);
        End;
    End;
  Writeln(' Кіл-У чисел = ',NumNumbers,
          '   Сума цифр = ',Sum);
End.
```

Виділюваний зі складу числа елемент *S*[*i*] є символом, а не цифрою. Тому для перетворення його в чисельне значення в програмі використовується процедура Val.

Приклад 15. У масиві рядків утримуються прізвища студентів. Згрупувати масив за алфавітом.

У таблиці ASCII українські букви записані за алфавітом. Пропуск має порядковий номер 32, значно менший порядкового номера будь-якої букви. Тому програма угруповання текстових рядків за алфавітом відрізняється від програми угруповання одновимірного масиву по зростанню лише тим, що тут замість порівняння й обміну місцями елементів числового масиву порівнюються і обмінюються місцями елементи масиву рядків.

Для угруповання масиву *Stud* у програмі StudGroup використовується метод прямого обміну (метод "пухирця").

```
Program StudGroup;
Const  Nmax = 100;
Type
   string30 = string[30];
   StringAr = array[1..Nmax] of string30;
Var
   i,                { параметр циклу }
   m,                { буферна змінна }
   n : byte;         { розмір масиву Stud }
   Cond : boolean;   { керуюча змінна }
      S : string30;  { буферний рядок }
   Stud : StringAr;  { масив рядків із прізвищами }
Begin
  Уведення  n, Stud
  Cond:=true; m:=n-1;
  While Cond do
    Begin
      Cond:=false;
      For i:=1 to m do
        If Stud[i]>Stud[i+1] then
          Begin
            S:=Stud[i]; Stud[i]:=Stud[i+1];
            Stud[i+1]:=S; Cond:=true;
          End;
      Dec(m);
    End;
  Печатка Stud
End.
```

Приклад 16. Відстань між двома словами рівної довжини - це кількість позицій, у яких різняться ці слова. Для заданого рядка тексту вказати номери й значення слів з максимальною відстанню між ними.

За умовою 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
```

У загальному випадку для пари (*i* - *j*), де *i, j* - номера слів у рядку, змінні *i* і *j* повинні змінюватися в наступних діапазонах:

$$i = 1..(n\text{-}1), \quad j = i+1..n .$$

У програмі Distance для перебору слів використовуються два цикли. У зовнішньому циклі, керованому булівською змінною *Cond*1, вибирається перше слово, у внутрішньому, керованому змінною *Cond*2, - друге слово порівнюваної пари слів. Обчислення відстані між словами, скопійованими в рядки *Sb*1 і *Sb*2, виконується, якщо кількість символів у порівнюваних словах однакова, тобто однакові довжини рядків *Sb*1 і *Sb*2. Для того, щоб велика й мала літери вважалися однією і тією же буквою, за допомогою функції UpCase виробляється перетворення малих латинських букв у великі букви.

```
Program Distance;
Var
   S,                { оброблюваний рядок }
   Sb1,Sb2,Sb3,Sb4,  { буф.рядок для аналізованих }
                     { слів }
   Sb1max,Sb2max     { слова з макс.відстанню }
        : string;    { між ними }
   i,                { параметр циклу }
   k1,k2,    { ліва й права границі першого слова }
   k3,k4,    { ліва й права границі другого слова }
   n1,n2,            { поточні номери слів }
   np1,np2,        { номера слів з макс.відстанню }
   d,dmax          { поточна й макс.відстані }
        : byte;
   Res,         { ознака наявності слів одинак.довжини }
   Cond1,       { змінна для керування }
                { зовнішнім циклом }
   Cond2        { те ж для внутрішнього циклу }
        : boolean;
Begin
  Readln(S);
  dmax:=0; Res:=false;
  k2:=0; Cond1:=true;
  n1:=0;
  While Cond1 do
    Begin
      k1:=NotSpace(S,k2+1);
      If k1=0 then
        Cond1:=false
      Else
        Begin
          k2:=Space(S,k1+1);
          If k2=0 then
            Cond1:=false
          Else
            Begin
              Inc(n1);
              Sb1:=Copy(S,k1,k2-k1);
```

```
                k4:=k2+1; Cond2:=true; n2:=n1;
                While Cond2 do
                  Begin
                    k3:=NotSpace(S,k4+1);
                    If k3=0 then
                      Cond2:=false
                    Else
                      Begin
                        k4:=Space(S,k3+1);
                        If k4=0 then
                          Begin
                            k4:=length(S)+1;
                            Cond2:=false;
                          End;
                        Inc(n2);
                        Sb2:=Copy(S,k3,k4-k3);
                        If length(Sb1) =
                               length(Sb2) then
                          Begin
                            Res:=true; d:=0;
                            Sb3:=Sb1; Sb4:=Sb2;
                            For i:=1 to length(Sb3) do
                              Sb3[i]:=UpCase(Sb3[i]);
                            For i:=1 to length(Sb4) do
                              Sb4[i]:=UpCase(Sb4[i]);
                            For i:=1 to length(Sb1) do
                              If Sb3[i]<>Sb4[i] then
                                Inc(d);
                            If d>dmax then
                              Begin
                                dmax:=d;
                                np1:=n1; np2:=n2;
                                Sb1max:=Sb1;
                                Sb2max:=Sb2;
                              End;
                          End;
                      End;
                  End;
              End;
          End;
      End;
    If not Res then
      Writeln('У рядку немає слів однакової довжини')
    Else
      Begin
        Writeln('dmax = ',dmax);
        Writeln('np1 = ',np1,'   Sb1max = ',Sb1max);
        Writeln('np2 = ',np2,'   Sb2max = ',Sb2max);
      End;
End.
```

Приклад виконання завдання.

Умова завдання. До складу вхідного тексту входять слова, що 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 представлене в наведеній нижче програмі TextWork.

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 і TextWork, повинна бути перевірена на системі тестів, у яких відбиваються різні комбінації вхідних даних. Якщо в процедурі Assign вказати конкретне ім'я зовнішнього файлу, наприклад,'InText.dat', то при переході від одного тесту до іншого необхідно попередньо набирати із клавіатури новий уміст файлу InText.dat або пересилати у файл InText.dat інший файл, що містить необхідний тест. Це сполучено з певними незручностями для користувача.

Більш прийнятним рішенням є організація такої роботи програми, при якій користувач міг би вказувати по запиту програми повністю 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 TextWord формується в рядку NameInFile і може приймати значення InText0.dat, InText1.dat, ..., InText9.dat, тобто всього допускається 10 файлів, що містять тестові вхідні дані. Імена файлів відрізняються лише одною цифрою, що умовно можна вважати номером вхідного файлу. Цей номер користувач повідомляє із клавіатури після відповідного запиту програми. Уведення вхідного тексту виконується з файлу InTextN.dat, де *N* - номер вхідного файлу. Змінна *n* визначає кількість рядків уведеного тексту. Уведений текст розміщується в масиві рядків *St* і виводиться на екран дисплея, а також, якщо це потрібно, друкується на принтері.

Таблиця результатів складається із чотирьох колонок: 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 привласнюється початкове нульове значення, а елементам масиву рядків TabWord - початкове порожнє значення. Якщо при аналізі вхідного тексту не буде виявлена яка-небудь буква, то це буде свідчити про її відсутність.

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 надалі використовується

рядок *AlphaBet*, у якому записані всі великі й малі літери латинського алфавіту. Рядок *AlphaBet* описаний у розділі **Const.**

Формування таблиці результатів і одночасне вилучення з тексту слів, у яких вперше зустрілася яка-небудь буква, виконується 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 виконується в циклі по параметру *i*, де *i* визначає номер рядка тексту.

Для і-го рядка спочатку визначається його довжина *l*, значення першого символу *ch* і положення цього символу *PosCh* у рядку-константі *AlphaBet*. Аналіз і-го рядка триває в циклі **While** доти, поки зберігає значення *true* булівська змінна *Cond*. У цьому циклі на першому етапі визначається номер початкової позиції *k*1 чергового слова тексту. Пошук першої букви цього слова виконується в циклі **While** доти, поки не буде виконана умова *PosCh* > 0 або поки не буде закінчено перегляд всього рядка (*k*1=*l*).

Якщо після закінчення даного циклу буде мати місце *PosCh* = 0, то це означає, що в і-ому рядку більше немає букв. Тоді змінній *Cond* привласнюється значення *false*, цикл "**While** *Cond* **do**" припиняє свою роботу і виконується перехід до аналізу наступного рядка тексту.

При *PosCh*>0 виконується пошук найближчого символу, що не є буквою. Позиція *k*2 цього символу визначає кінець чергового слова. Якщо після закінчення циклу **While** пошуку такого символу має місце *PosCh*<>0, то це означає, що остання буква даного слова розташована в останній позиції і-ого рядка. Тоді змінній *Cond* привласнюється значення false, що надалі приведе до переходу на (*i*+1)-ий рядок, а змінна *k*2 збільшується при цьому на 1 (*k*2 - це номер позиції після кінцевої букви даного слова). Змінній *Cond* привласнюється значення *false* також при *k*2 = l (кінцева буква слова перебуває в передостанній позиції рядка, роздільник - у його останній позиції).

Чергове слово, границі якого визначені значеннями змінних *k*1 і *k*2, переноситься в рядок *Slovo*; значення змінної *NumberWord* збільшується на 1. Після цього в циклі по параметру *j* проглядаються всі символи рядка *Slovo*. Змінна *k* визначає порядковий номер букви слова в латинському алфавіті. Якщо k-а буква ще не була виявлена у вхідному тексті (*TabNumberWord*[*k*]=0), то в елемент масиву *TabNumberWord*[*k*] заноситься номер слова *NumberWord*, в елемент *TabWord*[*k*] - значення цього слова, а змінна *KolLetter* збільшується на 1. Оскільки в даному слові вперше зустрілася одна з букв латинського алфавіту, то дане слово надалі повинне бути вилучене з тексту. Для цього змінній *CondDelete* привласнюється значення *true*.

Якщо після закінчення циклу по *j* має місце *CondDelete=true*, то слово з границями *k*1 і *k*2 вилучається з тексту, визначається нове значення довжини і-го рядка, а змінній *k*2 привласнюється значення *k*1.

Якщо на даний момент кількість відзначених букв *KolLetter* досягло значення 26 (кількість букв у латинському алфавіті), то це означає, що подальший перегляд вхідного тексту вже не має змісту. Тоді за допомогою оператора **Goto** виконується вихід за межі циклу по *i,* а на екран для контролю виводиться перетворений текст і таблиця результатів. Після цього таблиця результатів записується у вихідний файл.

Значення *Cond=true* на кінцевому етапі циклу "**While** *Cond* **do**" означає, що перегляд і-го рядка ще не закінчений. Тоді відповідним чином змінюються змінні *k*1, *ch* і *PosCh,* і аналіз і-го рядка триває.

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

В процесі перегляду вхідного тексту було вилучено з його складу деяку кількість слів, що привело до скорочення довжин рядків тексту. Для вирівнювання рядків тексту по правій границі необхідно в кожний з рядків, довжина якого менше оголошеної (у цьому випадку 66), перенести максимально можливу початкову ділянку наступного рядка. Природно, при такому переносі цілі десяткові числа, що входять до складу вхідного тексту, не повинні розділятися на дві частини. Слова відповідно до загальних методичних вказівок до даної контрольної роботи також не дозволяється розділяти знаком переносу. Тому з наступного рядка повинно бути взято таку початкову ділянку, яка закінчується роздільником, а не буквою або цифрою. Ця робота виконується процедурою RightBorder.

Для визначення приналежності аналізованого символу до роздільників використовується рядок *Separs,* описаний в розділі **Const**. У цьому рядку як роздільники перераховані пропуск, крапка, кома, крапка з комою, двокрапка. Для індикації початку й кінця чергового слова застосовуються функції SignBegin і SignEnd.

У 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 виконується в циклі **While** під керуванням змінної *CondText*.

Для формування нового рядка використовується змінна *S*1, куди до початку циклу перебору заноситься перший рядок тексту *St*. У змінну *S*2 заноситься черговий і-ий рядок оброблюваного тексту (початкове значення *i*=2). Якщо сумарна довжина *l*1+*l*2 рядків *S*1 і *S*2 менше 66, то до рядка *S*1

дописується вміст рядка *S*2, 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 $l1+l2 > 66$ відбувається частковий перенос слів з рядка *S*2 у рядок *S*1, що виконує цикл **While** під керуванням змінної **CondString.** Границі чергового слова в рядку *S*2 визначають змінні *k*1 і *k*2; значення $l = k2 - k1$ - це довжина слова. Якщо чергове слово може бути розміщене в *S*1 ($l1+l <= 66$), то воно переноситься в *S*1 і вилучається з рядка *S*2. Відношення $l1+l > 66$ означає, що рядок *S*1 уже заповнений. У цьому випадку цикл перегляду рядка *S*2 припиняє свою роботу, до значення змінної *is* додається 1, значення *S*1 переноситься в рядок *St*[*is*] нового тексту, а в *S*1 записується залишок рядка *S*2 і номер рядка *i* старого тексту збільшується на 1. При повторенні зовнішнього циклу в *S*2 буде занесений черговий рядок перетвореного масиву *St*.

Після закінчення зовнішнього циклу **While** останньому n-ому рядку нового тексту привласнюється значення рядка *S*1.

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

```
Program TextWork;
Uses Crt,Printer;
Const
   Nmax= 50;      { макс. кількість елементів у масиві }
                  { рядків St }
   Enter = 13;    { код клавіші Enter }
   Escape = 27;   { код клавіші Esc }
   PressKey = 'Натисніть клавішу Enter';
   AlphaBet = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'+
              'abcdefghijklmnopqrstuvwxyz';
                          { латинський 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 }
   KolDelete,          { кількість слів, що вилучаються, }
   PosCh       { позиція символу в рядку AlphaBet }
               { або в рядку Separs                 }
          : byte;
   NumberWord
        : integer;  { поточний номер слова в тексті }
   IndPrinter    { індикатор використання принтера }
        : boolean;
   ch,                 { символ тексту }
   Reply : char;       { символ відповіді }
                       {на запит програми }
   S1,S2,         { рядки для перетворення чисел }
```

```
   Slovo,       { 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            { масив номерів слів }
                            { таблиці результатів }
       : array[1..26] of integer;
   FileText : text;       { вхідний файл }
   FileOut                { вихідний файл }
            : file of String66;
{ ---------------------------------------------- }

Procedure WaitEnter;
{ Затримка виконання програми доти, }
{ поки не буде натиснута клавіша Enter        }
Var S:char;
Begin
  Repeat
    S:=ReadKey;
  Until ord(S) = Enter;
End { WaitEnter };
{ ---------------------------------------------- }

Procedure PrintString(X,Y:integer; S:string);
{ Печатка рядка S з позиції X рядка }
{ екрана з номером Y }
Begin
  GotoXY(X,Y);
  Write(S);
End { PrintString };
{ ---------------------------------------------- }

Procedure PrintKeyAndWaitEnter;
{ Друк рядка-константи PressKey з позиції 1 }
{ рядка екрана 25 і затримка виконання }
{ програми до натискання 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='Наст.сторінка - "Enter", кінець перегляду - '+
      '"Escape"';
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);
{ Друк на екрані 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;
{ Друк на екрані 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;
{ Друк на екрані вихідного файлу }
Var  i : integer;
    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 }
Var  i : integer;
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 };
{ ---------------------------------------------- }

Procedure MakeTabResult;
{ Формування таблиці результатів }
{ і вилучення відзначених слів   }
Var  i,j,l,k,k1,k2 : byte;
     Cond,
     CondDelete : boolean;
Begin
  NumberWord:=0; KolLetter:=0;
  For i:=1 to n do
    Begin
```

```
l:=length(St[i]);
Cond:=true; k1:=1;
ch:=St[i][k1]; PosCh:=Pos(ch,AlphaBet);
While Cond do
  Begin
    While (PosCh=0) and (k1<l) do
      Begin
        Inc(k1); ch:=St[i][k1];
        PosCh:=Pos(ch,AlphaBet);
      End;
    If PosCh=0 then
      Cond:=false
    Else
      Begin
        k2:=k1+1;
        If k2<=l then
          Begin
            ch:=St[i][k2];
            PosCh:=Pos(ch,AlphaBet);
          End;
        While (PosCh<>0) and (k2<l) do
          Begin
            Inc(k2); ch:=St[i][k2];
            PosCh:=Pos(ch,AlphaBet);
          End;
        If PosCh<>0 then
          Begin
            Inc(k2); Cond:=false
          End;
        If k2=l then
          Cond:=false;
        Slovo:=Copy(St[i],k1,k2-k1);
        Inc(NumberWord);
        CondDelete:=false;
        For j:=1 to k2-k1 do
          Begin
            ch:=Slovo[j]; k:=Pos(ch,AlphaBet);
            If k>26 then
              Dec(k,26);
            If TabNumberWord[k]=0 then
              Begin
                TabNumberWord[k]:=NumberWord;
                TabWord[k]:=Slovo;
                Inc(KolLetter);
                CondDelete:=true;
              End;
          End;
        If CondDelete then
          Begin
            Delete(St[i],k1,k2-k1);
            l:=length(St[i]);
            k2:=k1;
          End;
```

```
              If KolLetter=26 then
                Exit;
              If Cond then
                Begin
                  k1:=k2+1; ch:=St[i][k1];
                  PosCh:=Pos(ch,AlphaBet);
                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 в тексті 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]; PosCh:=Pos(ch,Separs);
            If PosCh>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 };
{ ---------------------------------------------- }

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 RightBorder;
{ Вирівнювання 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
                      Slovo:=Copy(S2,k1,k2-k1);
                      S1:=S1+' '+Slovo;
                      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

{ Установка відповідності між внутр. і зовн. файлами }
  ClrScr;
  Writeln('Укажіть номер вхідного файлу (0..9)');
  Read(NumberFile); Str(NumberFile:1,S1);
  NameInFile:=Concat('InText',S1,'.dat');
  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('Вихідний 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 в тексті 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 tt
rrr vbnmnm qwer,ttyui op 1234 ajdfj  hjkl  rxcv vbnm  pdfg rty ww
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   tt
rrr vbnmnm qwer,  1234 ajdfj     vbnm   rty ww
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 tt
rrr vbnmnm qwer, 1234 ajdfj vbnm rty ww
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 tt rrr vbnmnm qwer, 1234 ajdfj vbnm rty ww
fghju 6789 890 hjk fder vbmnd nbnvbfdf dfg n vcv cvvbi adfgh yuiop
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	20	pdfg
8	H	1	ddfjh
9	I	2	fjhiiii

```
10        J          1         ddfjh
11        K         17         hjkl
12        L         17         hjkl
13        M          8         vbnnm
14        N          8         vbnnm
15        O         15         op
16        P         15         op
17        Q          9         qwe
18        R          4         rty
19        S          0
20        T          4         rty
21        U         14         ttyui
22        V          3         AADCv
23        W          9         qwe
24        X         18         rxcv
25        Y          4         rty
26        Z         28         iyutzz

        П Е Р Е Т В О Р Е Н И Й    Т Е К С Т
5678,. 8765 rty acd tt rrr vbnmnm qwer, 1234 ajdfj vbnm rty ww
fghju 6789 890 hjk fder vbmnd nbnvbfdf dfg n vcv cvvbi adfgh yuiop
nbg nmmm tyyui ABCabc hgJG tghnbgtyhbn lkjhhttvbb uyhfggb ijhghfgw
kknbgfghj, ukjhg 654 oii 786, lk7mnhgfgg
```

В А Р І А Н Т И З А В Д А Н Ь

З а г а л ь н і з а у в а ж е н н я

1. Вхідний текст - це масив, що складається не менш ніж з 10 рядків.
2. Уведення вхідних даних повинно здійснюватися з текстового файлу.
3. Рядки перетвореного тексту повинні бути вирівняні таким чином, щоб кожна з них містила максимально можливу цілу кількість слів.
4. Перетворений текст необхідно записати в типізований файл. Вивід на екран і на принтер результатів рішення завдання повинно виконуватися із цього файлу.

---

1. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. Перенести у вихідний файл із вхідного тексту слова, що починаються й закінчуються голосною буквою.

---

2. Вхідний текст містить список цілих десяткових чисел, розділених комами. Перенести у вихідний файл числа-паліндроми, тобто. числа, які читаються однаково зліва направо і справа наліво (наприклад, 4554, 78987, 1 і

т.п.). Якщо яке-небудь із вхідних чисел містить незначащі нулі (наприклад, 0038783), то такі нулі повинні бути попередньо вилучені.

---

3. Вхідний текст містить два речення, кожне з яких закінчується крапкою. Кожне речення може займати кілька рядків, причому в останньому рядку першого речення може починатися друге речення. Визначити набір символів, що повинен бути доданий до першого речення, щоб з його складу можна було сконструювати друг речення.

---

4. Вхідний текст містить список ідентифікаторів. Елементи списку розділені між собою комами, ліворуч і(або) праворуч від коми можуть бути один або кілька пропусків. Перенести в перетворений текст список помилкових ідентифікаторів, розділивши їх між собою комами.

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

---

5. У вхідному тексті записані цілі десяткові числа, розділені між собою одним або декількома пропусками. Перетворити текст таким чином, щоб у кожному рядку спочатку були записані всі вхідні в неї додатні числа, а потім - всі від'ємні числа, причому відносна послідовність чисел у кожній із двох груп не повинна відрізнятися від вхідної. Числа в рядках нового тексту розділяти комами.

---

6. У вхідний текст входять слова, що складаються з латинських букв, цілі десяткові числа і роздільники (пропуск, кома, крапка, крапка з комою, двокрапка). Вилучити з тексту всі слова з непарними порядковими номерами і переставити у зворотному порядку букви слів з парними порядковими номерами.

Примітка. Нумерація слів повинна бути умовно зроблена по всьому тексту, представленому масивом рядків, а не по кожному рядку окремо.

---

7. Слова вхідного тексту складаються тільки з малих і великих латинських букв. Слова розділені між собою одним або декількома пропусками. У кожному слові розташувати вхідні в його склад букви за алфавітом, причому для однакових букв старшої вважається велика буква. Наприклад, для слова 'AXbzaTbB' одержимо 'AaBbbTXz'. Слова, довжина яких перевищує 10 символів, вилучити із тексту.

---

8. Слова в тексті розділені одним з наступних символів: пропуск, крапка, кома, крапка з комою, двокрапка, причому 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*.

---

9. Перенести в перетворений текст слова вхідного тексту, що містять подвоєні голосні (але не слова, що містять групу із трьох або більше підряд розташованих однакових голосних).

---

10. У вхідному тексті міститься список арифметичних констант у формі десяткових чисел із плаваючою комою, причому їхні мантиси нормалізовані. Елементи списку розділені між собою одним або декількома пропусками. Перенести в перетворений текст список додатних констант, значення порядку яких понад 1. Наприклад, у списку 0.12E+12 -0.677E37 0.234E-34 0.87E3 0.6E-5 0.6E+8 такими константами є 0.12E+12, 0.87E3, 0.6E+8. Елементи нового списку розділити між собою комами.

---

11. Текст містить слова, записані латинським буквами, і цілі десяткові числа. Як роздільники використовуються пропуск і кома. Вилучити з тексту всі числа, а в кожному з його слів переставити букви у зворотному порядку.

---

12. Слова вхідного тексту переставити у зворотному порядку.

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

---

13. Текст містить на українській мові список російських прізвищ чоловічого роду. Перетворити список таким чином, щоб спочатку йшли всі прізвища із закінченням -єв, потім - прізвища із закінченням -ін, у третій групі - прізвища із закінченням -ов, в останній групі - всі інші прізвища. Відносна послідовність прізвищ у кожній групі повинна бути та ж, що й у вхідному списку.

---

14. У вхідному тексті суцільним потоком записані групи букв і групи цифр (наприклад, '4567abcrg675hgrfdbn87359sdfgghhj456732'). При цьому кожний черговий рядок вважається продовженням попереднього рядка. Відокремити кожну групу одним пропуском від сусідніх груп. Визначити кількість груп цифр, кількість груп букв, а також кількість різних букв і цифр, використаних у тексті.

---

15. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з 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.

---

16. Вхідний текст містить слова і цілі десяткові числа. У кожному числі переставити його цифри у зворотному порядку. Вилучити в числі незначащі нулі, якщо вони утворилися після перестановки цифр. Наприклад, при перестановці цифр числа 3508000 потрібно одержати 8035, а не 0008035.

---

17. Перенести у вихідний файл всі слова вхідного тексту, які складаються з тих же букв, що й перше слово тексту (поза залежністю від кількості повторень кожної букви).

---

18. У вхідному тексті міститься список десяткових констант із фіксованою крапкою. Елементи списку розділені комами. Перенести у вихідний файл список неправильних констант, розділивши їх на групи по типу помилки (використання неприпустимих символів, наявність в одній константі декількох поділяючих крапок, наявність більш ніж одного знака "+" або "-", положення знака "+" або "-" у середині або наприкінці константи і т.п.). Врахувати, що в окремому випадку та сама помилкова константа може одночасно перебувати в декількох групах.

---

19. У вхідному тексті як роздільники використовуються пропуск, кома, крапка, крапка з комою, двокрапка. Перенести у вихідний файл слова-паліндроми, тобто слова, які однаково читаються зліва направо і справа наліво. Елементи нового списку розділити комами.

---

20. У вхідний текст входять слова і цілі десяткові числа. Як роздільники використовуються пропуск, крапка, кома, крапка з комою, двокрапка. Замінити кожне число сумою його цифр.

---

21. У вхідному тексті міститься список арифметичних констант у формі цілих десяткових чисел і у формі десяткових чисел, ціла і дробова частини яких розділені крапкою. Зменшити значення кожної константи в 10 разів. Якщо вхідна константа не має дробової частини й остання її цифра дорівнює нулю,

вилучити цю цифру; якщо остання цифра цілої константи не дорівнює нулю, вставити крапку перед останньою цифрою; якщо константа має дробову частину, переставити поділяючу крапку на один розряд уліво. При установці або зміні положення поділяючої крапки необхідно враховувати кількість цифр у цілій частині константи. Окремо повинен аналізуватися також випадок, коли вхідна константа дорівнює нулю.

---

22. Для запису слів вхідного тексту використовуються великі і малі латинські букви. Слова розділені між собою одним або декількома пропусками. Потрібно в кожному слові вилучити символи, які вже зустрічалися в цьому слові. Врахувати, що великі і малі літери в словах вважаються еквівалентними. Слова в перетвореному тексті розділяти одним пропуском.

---

23. У вхідному тексті є кілька пар відкриваючих і закриваючих дужок. Перенести в перетворений текст фрагменти вхідного тексту, що знаходяться у всіх внутрішніх дужках. Якщо в перенесених фрагментах є пропуски, їх необхідно вилучити; фрагменти розділити між собою двома пропусками. 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'.

Врахувати, що перенесений текст може розташовуватися не на одному рядку.

---

24. Вхідний текст містить список десяткових чисел, у який входять цілі десяткові числа і десяткові числа з фіксованою комою (ціла і дробова частини числа розділені крапкою). Деякі числа можуть містити в явному вигляді знак "+". Вилучити з тексту знак '+' перед додатними числами, а також від'ємні числа, що не перевищують по модулю значення 10. Визначити в новому тексті кількості додатних, від'ємних і нульових чисел.

---

25. Вхідний текст містить список десяткових констант із фіксованою крапкою, причому кожна константа має явно виражені цілу і дробову частини. У кожній константі відкинути незначащі нулі в цілій і дробовій частинах, якщо такі нулі входять до складу константи. Якщо дробова частъ константи нульова, відкинути також поділяючу крапку. Врахувати, що в окремому випадку константа може мати вигляд 0.0.

---

26. Слова вхідного тексту розділені одним або декількома пропусками. При записі слів допускаються переноси. Перетворити текст таким чином, щоб

кожне його слово перебувало в одному рядку тексту (тобто позбутися від переносів). Слова в новому тексті повинні бути розділені лише одним пропуском. На початку першого рядка тексту встановити п'ять пропусків, на початку інших рядків пропуски повинні бути відсутніми.

---

27. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. Перенести у вихідний файл із вхідного тексту слова, що починаються із голосної букви, причому ця буква в слові більше не повторюється.

---

28. Слова в тексті розділені між собою тільки одним роздільником, у якості якого використовується пропуск, кома або крапка. На початку кожного абзацу передбачається п'ять пропусків. Рядки тексту в загальному випадку не вирівняні по правому краї.

Відредагувати текст, вирівнявши по правому краї кожний його рядок, крім тих рядків, які є останніми в абзацах. При вирівнюванні додавати в проміжки між словами необхідну кількість пропусків, причому всі проміжки повинні відрізнятися між собою не більше ніж на один пропуск.

---

29. Вихідний текст українською мовою складається зі слів, чисел і роздільників (пропуск, крапка, кома, крапка з комою, двокрапка), причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. Замінити числа, що складаються з однієї або двох цифр без знака, українськими словами. Наприклад,

замість 6 записати " шість",
замість 14 - "чотирнадцять",
замість 41 - "сорок один" і т.д.

---

30. До складу тексту входять слова, записані латинськими буквами, і цілі десяткові числа. Слова і числа розділені між собою одним або декількома пропусками. Потрібно в кожному рядку тексту розташувати спочатку всі вхідні в його склад числа, а потім - слова, причому відносний порядок чисел і слів у своїх підгрупах не повинен відрізнятися від вхідного. Слова і числа розділяти між собою комами. Наприкінці групи чисел записати крапку з комою. Врахувати окремі випадки (у рядку немає чисел; до складу рядка входять тільки числа).

---

31. У вхідному тексті записано список арифметичних констант у формі цілих десяткових чисел і у формі десяткових чисел, ціла й дробова частини яких розділені крапкою. Збільшити значення кожної константи в 10 разів. Якщо вхідна константа не має дробової частини, дописати до неї цифру 0; якщо в дробовій частині константи міститься одна цифра, то записати цю цифру на місце поділяючої крапки; якщо дробова частина константи має понад одну цифру, то першу цифру дробової частини і поділяючу крапку поміняти місцями.

---

32. Слова в тексті розділені одним з наступних символів: пропуск, крапка, кома, крапка з комою, двокрапка, причому кожний із цих роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. Вилучити з тексту слова, що складаються з однієї букви, після чого визначити середню довжину S слів, що залишилися, і сформувати список слів, довжина яких перевищує значення S. Слова в новому списку повинні розділятися комами.

---

33. У вихідному тексті записано список арифметичних констант у формі десяткових чисел з фіксованою комою. Елементи списку розділені між собою одним або декількома пропусками. Якщо дробова частина константи має більше двох цифр, виконати її округлення до двох цифр. Елементи нового списку констант розділити комами.

---

34. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. Перенести у вихідний файл із вхідного тексту слова, які починаються й закінчуються однієї й тією же буквою, причому ця буква в слові більше не повторюється.

---

35. У вхідному тексті українською мовою зустрічаються числа без знака, після яких стоїть слово "грн.". Замінити кожний такий вартісний показник значенням в "грн." і "коп.", причому для запису значення в "коп." завжди використовувати дві цифри. Приклади:

3.8 грн. = 3 грн. 80 коп.; 0.56 грн. = 0 грн. 56 коп.; 28 грн. = 28 грн 00 коп.

---

36. Вхідний текст містить список цілих десяткових чисел, причому знак числа записаний після останньої цифри числа. Числа розділені між собою одним або декількома пропусками. Перетворити список таким чином, щоб для

від’ємних чисел знак був записаний перед першою цифрою числа, а для додатних чисел знак не був зазначений. Наприклад, для списку

'3867+ 564- 12345- 0+ 11- 304+'

одержимо

'3867 -564 -12345 0 -11 304'.

Числа в новому списку повинні бути розділені комами.

---

37. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пробілами. Перенести у вихідний файл із вхідного тексту слова, у яких є більше двох підряд розташованих приголосних букв.

---

38. У вхідному тексті записано список арифметичних констант у формі десяткових чисел з фіксованою крапкою. Елементи списку розділені між собою одним або декількома пропусками. Перенести в перетворений текст список констант, ціла частина яких складається більш ніж із трьох цифр. Елементи нового списку розділити комами.

---

39. У вхідний текст входять слова, що складаються із великих і малих латинських букв, цілі десяткові числа і роздільники (пропуск, кома, крапка, крапка з комою, двокрапка). Перетворити даний текст, вилучивши з нього слова, цілком складені із входжень не більш ніж двох букв (наприклад, 'Akka', 'RrrRRr', 'SttTs' і т.п.). При цьому прописні й малі літери вважаються еквівалентними. Елементи отриманого списку слів розділити комами.

---

40. Текст містить список десяткових чисел з плаваючою комою. Числа розділені між собою одним або декількома пропусками. Вилучити з тексту числа, мантиса яких не перевищує по модулю значення 0.5 і вставитъ знак "+" для тих чисел, які не мають явно записаного знака перед мантисою або перед порядком. Для нового списку як роздільник використовувати кому. Наприклад, для тексту

'3.52E-12 0.14E+3 7.77E14 -3.62E4 0.15E1 6.666E13 -1.2E-1 0.45E5'

одержимо

'+3.52E-12,+7.77E+14,-3.62E+4,+6.666E+13,-1.2E-1'.

---

41. Для поділу слів у вхідному тексті українською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. У складі тексту можуть також перебувати

цілі десяткові числа. Перенести у вихідний файл із вхідного тексту слова, у яких відсутнє подвоєння букв.

---

42. До складу кожного слова вхідного тексту можуть входити латинські букви і цифри. Якщо на початку слова стоять цифри, то переставити їх циклічно в кінець слова для того, щоб кожне слово починалося з букви. Перша буква кожного слова повинна бути великою, інші - малими. Наприклад, для слова '367abCD8Fg' одержимо 'Abcd8fg367'.

---

43. Для поділу слів у вхідному тексті російською мовою використовуються пропуск, крапка, кома, крапка з комою, двокрапка, тире, причому кожний з роздільників може бути оточений ліворуч і(або) праворуч одним або декількома пропусками. З кожного рядка тексту перенести у вихідний файл пари слів, що складаються з тих самих букв (наприклад, товар і автор, весна і навес, салат і атлас, слово і волос і т.п.). Між словами кожної пари записувати союз "і", пари слів розділяти комами.

Примітка. У виділеній парі слів кількість кожної з букв повинне бути однаковим. У цьому змісті слова "насос" і "нанос" не відповідають умові завдання.

---

44. У вхідний текст входять слова, що складаються із великих і малих латинських букв, і цілі десяткові числа. Слова і числа розділені між собою одним або декількома пропусками. У кожному рядку тексту вилучити слова, які раніше зустрічалися в цьому ж рядку. При цьому врахувати, що великі і малі літери в словах вважаються еквівалентними.

---

45. У вхідному тексті записані цілі десяткові числа, розділені між собою одним або декількома пропусками. Перетворити текст таким чином, щоб у кожному рядку спочатку були записані всі вхідні в неї парні, а потім - всі непарні числа, причому відносна послідовність чисел у кожній із двох груп не повинна відрізнятися від вхідної. Числа в рядках перетвореного тексту повинні розділятися між собою комою.

---

46. Для запису слів вхідного тексту використані великі і малі латинські букви. Слова розділені між собою одним або декількома пропусками. Перенести в перетворений текст слова, що повторюються в межах одного рядка. Кожне таке слово в перетвореному тексті повинне бути записане один раз, а після нього в дужках повинне бути зазначено кількість повторень. Врахувати, що великі й малі літери в словах вважаються еквівалентними. У

новому тексті перша буква кожного слова повинна бути великою, інші букви - малими.

---

47. У кожному рядку вхідного тексту вилучити пари слів, з яких одне є оберненням іншого (наприклад, 'огород' і 'дорого').

---

48. Текст містить список цілих десяткових чисел, розділених комами. Частина чисел може мати знак "+" або "-". Замінити кожну пару суміжних чисел їхньою напівсумою, що округлена до найближчого цілого значення. Якщо в рядку непарна кількість чисел, то для формування пари береться останнє число даного рядка і перше число наступного рядка.

---

49. Вхідний текст містить кілька арифметичних виразів, до складу яких входять ідентифікатори простих змінних, цілі десяткові числа, знаки арифметичних операцій і дужки. Вирази розділені між собою крапкою з комою. Між елементами виразів можуть бути пропуски. У кожному рядку може бути кілька виразів. Проаналізувати в кожному виразі баланс відкриваючих і закриваючих дужок і у випадку його порушення вилучити надлишкові зовнішні дужки. Пропуски між елементами виразу вилучити. У перетвореному тексті після кожної крапки з комою записати один пропуск. Наприклад, для рядка

r-2 *(a + b5*(k-1); St / (7 - 4* (kt +5)))- 6 * c)+1;

одержимо

r-2*a+b5*(k-1); St/(7-4*(kt+5))-6*c+1;

---

50. Вхідний текст містить список цілих десяткових чисел, розділених між собою одним або декількома пропусками. Перед першою цифрою числа може бути записаний знак "+" або "-". У кожному рядку вихідного тексту записати спочатку числа, що починаються із цифри 1, потім - із цифри 2, 3, 4 і т.д., причому відносна послідовність чисел у кожній групі повинна бути такою ж, як і у вхідному тексті. Якщо яке-небудь із вхідних чисел містить незначні нулі, то такі нулі повинні бути вилучені. Якщо в рядку є число 0, то це число повинне бути першим у рядку. Числа в перетвореному тексті розділяти між собою одним пропуском.
