МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕТОДИЧНІ ВКАЗІВКИ І ЗАВДАННЯ ДО КОНТРОЛЬНОЇ РОБОТИ ПО КУРСУ ПРОГРАМУВАННЯ

ЧАСТИНА 2

(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 2 (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. – 43 с.

Приведені методичні вказівки до контрольної роботи № 2 по курсу програмування для студентів-заочників спеціальності 7.091501. Контрольна робота містить завдання по темам «Обробка матриць» і «Використання процедур і функцій». Для виконання контрольної роботи по кожній темі 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.

ЗАГАЛЬНІ МЕТОДИЧНІ ВКАЗІВКИ

Контрольна робота №2 для студентів-заочників спеціальності 7.09.15.01 "Комп'ютерні та інтелектуальні системи і мережі" включає в себе два завдання, які зорієнтовані на розробку і виконання програм на мові Турбо Паскаль.

Контрольна робота, що подається студентом на кафедру, по кожному завданню повинна містити:

- умову задачі;
- стислий опис алгоритму і програми;
- лістинг програми;
- результати рішення задачі.

В методичних вказівках до кожного завдання можуть бути наведені додаткові вимоги до описової частини контрольної роботи.

Кожне завдання містить 50 варіантів задач. Вибір варіанту проводиться по двох останніх цифрах номеру залікової книжки, збільшених на 1. Якщо одержане число перевищує значення 50, то це число зменшується на 50.

Матеріал контрольної роботи повинен бути приведений на аркушах паперу формату А4. При цьому на першому листі треба написати:

Донецький національний технічний університет
Кафедра електронних обчислювальних машин

Контрольна робота № 2
по курсу "Програмування"

студента групи (шифр групи)
Прізвище та ініціали

Номер залікової книжки .....
Домашня адреса: .....

Текст програми (лістинг) і результати рішення задачі повинні бути надруковані на принтері.

Тексти методичних вказівок містяться не тільки в посібниках, виданих типографським засобом. Вони записані також на комп'ютерах в дісплейних класах кафедри ЕОМ. Студентам рекомендується переписати тексти методичних вказівок на дискету і використовувати наведені в них програмні приклади як основу для розробки свого варіанту виконання відповідного завдання. В першу чергу ця рекомендація відноситься до сервісних процедур і функцій. Включення їх в свою програму шляхом копіювання файлу принаймні економить час, який нераціонально витрачається при наборі цих процедур з клавіатури.

З А В Д А Н Н Я N 1

ОБРОБКА МАТРИЦЬ

Методичні вказівки

Мета роботи - практично освоїти засоби розробки алгоритмів і засоби програмної реалізації задач обробки багатовимірних масивів.

В кожному з варіантів завдання № 1 необхідно скласти програму, що передбачає:

- введення з текстового файлу вхідної матриці;
- друк вхідних даних;
- обробку матриці відповідно з умовою задачі;
- виведення результатів.

Виведення результатів повинно бути організовано як на екран дісплею, так і на принтер. Для організації виведення результатів рекомендується використовувати процедури ScreenMatrix і PrinterMatrix, наведені в прикладі.

Введення матриці з текстового файлу, на відміну від одновимірного масиву, не можна організувати по ознаці кінця файлу (по значенню функції SeekEof). При введенні матриці m x n елементів з використанням вказаної ознаки програма може визначити лише загальну кількість елементів матриці, але вона не в змозі окремо вказати значення параметрів m і n. Тому рекомендується при формуванні текстового файлу в першому його рядку записати значення m і n (для квадратної матриці - тільки значення n), а в інших рядках - значення елементів матриці. Після введення з файлу змінних m і n введення елементів матриці відбувається в циклі for, як це зроблено в прикладі.

Процедура WaitEscape, що міститься в прикладі, використовується для зупинки роботи програми до того часу, доки не буде натиснута клавіша Esc (ця клавіша має код 27).

Приклад виконання завдання.

Умова задачі. В заданій квадратній матриці дійсних чисел зсунути кожний її рядок циклічно ліворуч або праворуч таким чином, щоб максимальний елемент рядка був розміщений на головній діагоналі. Обчислити слід матриці до і після її перетворення.

Циклічний зсув рядка припускає таку перестановку його елементів, при якому не порушується їхнє вхідне відносне розташування.

Нехай, наприклад, третій рядок матриці має вигляд

5.8 -4.4 10.0 8.7 14.6 11.9

Тут максимальний елемент a[3,5] розміщений праворуч на дві позиції від елементу головної діагоналі a[3,3]. Отже, необхідно двічі виконати циклічний

зсув рядка ліворуч, після чого він набуде вигляду

10.0 8.7 14.6 11.9 5.8 -4.4

Слід матриці - це сума елементів її головної діагоналі.

Завдання реалізоване в програмі Task2_1, яка наведена нижче.

При обробці матриці в кожному її і-ому рядку визначається максимальний елемент *Amax* і його індекс *jmax*, після чого обчислюється параметр *k = i-jmax*, що вказує положення елементу *Amax* по відношенню до головної діагоналі матриці. Якщо $k > 0$ (максимальний елемент знаходиться ліворуч від елементу головної діагоналі), то виконується циклічний зсув рядка праворуч до тих пір, доки параметр *k* не прийме нульове значення (в кожному циклі значення k зменшується на 1). Останнє означає, що максимальний елемент переставлений на місце елементу головної діагоналі. При $k < 0$ циклічний зсув виконується ліворуч.

В програмі Task2_1 передбачено дві можливості виведення результатів: на екран дісплея та на принтер. Необхідність використання принтера визначається відповіддю користувача на відповідний запит програми при черговому запуску програми.

Процедура ScreenMatrix, що використовується для виведення матриці на екран дісплею, передбачає, що для розміщення матриці достатньо однієї сторінки екрану. Якщо це не так, то ця процедура повинна бути доповнена фрагментом, що забезпечує управління "перегортанням" сторінок екрану. Приклад такого доповнення для виведення одновимірних масивів наведений в методичних вказівках до завдання № 2.

```
Program Task2_1;
Uses Crt, Printer;
Const Nmax=30;
Type Matrix=array [1..Nmax, 1..Nmax] of real;
Var
   i,j,           { параметри циклу }
   n,             { розмір матриці }
   jmax : byte;   { позиція макс. елементу в рядку }
   k : shortint; { різниця між положеннями Amax }
                  { і елементу a[i,i] }
   IndPrinter : boolean; {індикатор використання принтера}
   ch : char;     { символ натиснутої клавіші }
   Amax,          { максимальний елемент в рядку }
   Buf,           { буферна змінна }
   Trace : real; { слід матриці }
   A : Matrix;    { матриця, що обробляється }
   F  : text;     { вхідний файл }
{ ------------------------------------------------------ }
Procedure WaitEscape;
{ Зупинення програми до натискування клавіші Esc }
Var ch : char;
```

```
Begin
  Repeat
    ch:=ReadKey;
  Until ord(ch)=27;
End { WaitEscape };
{ ------------------------------------------------- }
Procedure ScreenMatrix;
{ Виведення матриці на екран дісплею }
Var i,j,k : byte;
Begin
  For i:=1 to n do
    Begin
      k:=0;
      For j:=1 to n do
        Begin
          Inc (k);
          If  k<5 then
            Write(a[i,j]:8:2,'  ')
          Else
            Begin
              k:=0;
              Writeln(a[i,j]:8:2,'  ')
            End
        End;
      If k>0 then Writeln;
    End;
  If k>0 then Writeln;
End { ScreenMatrix };
{ ------------------------------------------------- }
Procedure PrinterMatrix;
{ Виведення матриці на принтер }
Var i,j,k : byte;
Begin
  For i:=1 to n do
    Begin
      k:=0;
      For j:=1 to n do
        Begin
          Inc(k);
          If  k<5 then
            Write(Lst,a[i,j]:8:2,'  ')
          Else
            Begin
              k:=0;
```

```
            Writeln(Lst,a[i,j]:8:2,'  ')
          End
        End;
      If k>0 then Writeln(Lst);
    End;
  If k>0 then Writeln(Lst);
End { PrinterMatrix };
{ ------------------------------------------------ }
Procedure TraceMatrix;
{ Обчислення сліду матриці }
Var i : byte;
Begin
  Trace:=0;
  For i:=1 to n do
    Trace:=Trace+a[i,i];
End { TraceMatrix };
{ ------------------------------------------------- }
Begin

{ Встановлення відповідності між внутрішнім і зовнішнім
файлами }
  Assign(F,'Matrix.dat');

{ Відкриття вхідного файлу }
  Reset(F);

{ Запит про використання принтера }
  ClrScr;
  Writeln('Чи буде використаний принтер (Так, Ні)?');
  ch:=ReadKey;
  If  ch in ['T','т','N','n'] then
    IndPrinter:=true
  Else
    IndPrinter:=false;

{ Введення і друк вхідних даних }
  Read(F, n);
  For i:=1 to n do
    For j:=1 to n do
      Read(F,a[i,j]);
  Writeln('Вхідна матриця');
  Writeln('n=',n);
  ScreenMatrix;
  If  IndPrinter then
```

```
    Begin
      Writeln(Lst,'Вхідна матриця');
      Writeln(Lst,'n=',n);
      PrinterMatrix;
    End;

{ Закриття вхідного файлу }
  Close(F);

{ Обчислення сліду вхідної матриці }
  TraceMatrix;
  Writeln('Слід вхідної матриці Trace =  ',Trace:8:2);
  If  IndPrinter then
     Writeln(Lst,'Слід вхідної матриці Trace=',Trace:8:2);
  WaitEscape;

{ Перетворення вхідної матриці }
  ClrScr;
  For i:=1 to n do
    Begin
      Amax:=a[i,1]; jmax:=1;
      For j:=2 to n do
        If  a[i,j]>Amax then
          Begin
            Amax:=a[i,j]; jmax:=j
          End;
      k:=i-jmax;
      If  k>0 then
        While k>0 do
          Begin
            Buf:=a[i,n];
            For j:=n downto 2 do
              a[i,j]:=a[i,j-1];
              a[i,1]:=Buf; Dec(k);
          End
        Else
          If  k<0 then
            While k<0 do
              Begin
                Buf:=a[i,1];
                For j:=1 to n-1 do
                  a[i,j]:=a[i,j+1];
                a[i,n]:=Buf; Inc(k);
              End;
```

```
    End;
  Writeln ('Перетворена матриця');
  ScreenMatrix;
  If IndPrinter then
    Begin
      Writeln(Lst);
      Writeln(Lst, 'Перетворена матриця');
      PrinterMatrix;
    End;

{ Обчислення сліду перетвореної матриці }
  TraceMatrix;
  Writeln ('Слід перетвореної матриці Trace = ',Trace:8:2);
  If  IndPrinter then
    Writeln(Lst,'Слід перетвореної матриці Trace=',
            Trace:8:2);
  WaitEscape;
End.
```

Результати роботи програми:

Вхідна матриця
n=6

1.00	0.00	0.00	3.00	13.00
23.00				
5.00	55.00	15.00	7.00	17.00
0.00				
9.00	19.00	0.00	81.00	88.00
18.00				
6.00	16.00	66.00	33.00	21.00
12.00				
99.00	15.00	44.00	0.00	13.00
11.00				
10.00	11.00	12.00	17.00	23.00
14.00				

Слід вхідної матриці Trace=116.00

Перетворена матриця

23.00	1.00	10.00	0.00	3.00
13.00				
5.00	55.00	15.00	7.00	17.00
0.00				
0.00	81.00	88.00	18.00	9.00

19.00
12.00 6.00 16.00 66.00 33.00
21.00
44.00 0.00 13.00 11.00 99.00
15.00
14.00 10.00 11.00 12.00 17.00
23.00

Слід перетвореної матриці Trace=354.00

ВАРІАНТИ ЗАВДАННЯ № 1

---

1. Задана квадратна матриця. Переставити у зворотному порядку елементи тих стовбців матриці, що розміщені нижче її головної діагоналі.

---

2. В кожному рядку прямокутної матриці від'ємним елементам присвоїти нульове значення. Після цього перенести всі додатні елементи на початок рядка в порядку їхнього вхідного відносного розташування.

---

3. Елементи кожного рядка прямокутної матриці зсунути циклічно праворуч, не порушуючи при цьому положення максимального елемента даного рядка.

Наприклад, для рядка 8 12 -6 14 11 -3 7 9 отримаємо

9 8 12 14 -6 11 –3 7.

---

4. Для кожного стовбця прямокутної цілочисельної матриці обчислити суму елементів, що в нього входять, і визначити, чи є в матриці стовбці з однаковою сумою. Обчислити кількість пар таких стовбців.

---

5. Виконати поточне згладження кожного рядка прямокутної матриці і визначити максимальне відхилення її елементів від середнього арифметичного значення даного рядка до і після згладження.

Примітка. При поточному згладженні масиву *x1, x2,..., xn* кожний *j*-ий елемент масиву (*j* = 2, 3,..., *n*-1) замінюється середнім арифметичним значенням елементів з індексами *j*-1, *j*, j+1.

---

6. В кожному рядку прямокутної матриці перенести максимальний елемент в останню позицію рядка, зсунувши при цьому ліворуч розташовані після нього елементи. Врахувати окремий випадок, коли максимальний елемент вже знаходиться в останній позиції рядка.

Приклад. Рядок 5 18 21 12 10 24 13 17 8 10
після перетворення буде мати вигляд:
5 18 21 12 10 13 17 8 10 24.

---

7. В прямокутній матриці кожний нульовий елемент замінити середнім арифметичним значенням ненульових елементів того рядка, в якому розміщений цей елемент. Врахувати випадок, коли всі елементи рядка нульові.

---

8. Дана квадратна матриця. Якщо в її трикутній частині, розташованій вище побічної діагоналі, є нульові елементи, то замінити кожний з них мінімальним (але відмінним від нуля) значенням елементів стовбця, в якому розміщений нульовий елемент.

Примітка. Побічна діагональ проходить від лівого нижнього до правого верхнього елементів матриці.

---

9. Для кожного рядка прямокутної цілочисельної матриці визначити суму її додатніх елементів, а після цього переставити рядки в порядку зменшення цих сум.

---

10. Визначити, чи є в прямокутній матриці лінійно залежні рядки і підрахувати кількість пар таких рядків.

Примітка. Два рядки матриці лінійно залежні, якщо один з них можна отримати з іншого множенням на постійний коефіцієнт.

---

11. Для кожного стовбця прямокутної матриці, елементами якої є цілі додатні числа, визначити суму елементів, що входять до нього, і, якщо вона непарна, додати одиницю до значення останнього елементу даного стовбця. Після цього згрупувати елементи стовбця в порядку їх зменшення.

---

12. Серед діагоналей квадратної матриці, паралельних головній і розташованих нижче неї, знайти таку, сума модулів елементів якої максимальна у порівнянні з іншими діагоналями.

---

13. В прямокутній матриці розглянути квадратні підматриці вимірністю 1, 2, 3,..., причому для всіх підматриць лівим верхнім елементом є елемент вхідної матриці з індексами (1,1). Визначити номер підматриці, середнє арифметичне елементів якої має найбільше значення.

---

14. В прямокутній матриці знайти номери стовбців, що містять відповідно максимальну і мінімальну кількість від'ємних елементів, після чого обміняти їх місцями. Врахувати випадки, коли: в матриці немає від′ємних елементів; лише один стовбець матриці містить такі елементи; лише два стовбця містять від'ємні елементи, але їхня кількість однакова.

---

15. В прямокутній матриці визначити кількість рядків, елементи яких повністю упорядковані по зростанню, після чого перенести ці рядки в початкову частину матриці, зберігши вхідне відносне розташування.

Вказівка. Якщо при перегляді елементів рядка знайдена пара елементів, які порушують його упорядкованість, подальший аналіз елементів рядка припинити.

---

16. В кожному стовбці прямокутної матриці є по крайній мірі один нульовий елемент. Замінити в кожному стовбці останній нульовий елемент таким значенням, щоб сума елементів стовбця дорівнювала нулю. Сформувати вектор, *i*-та компонента якого дорівнює різниці між сумою елементів *i*-го рядка (не стовбця!) до перетворення матриці і сумою елементів цього ж рядка після перетворення.

---

17. В квадратній матриці визначити суму додатних елементів, розташованих над головною діагоналлю, і суму від'ємних елементів, розташованих під головною діагоналлю. Якщо перша з них перевищує по модулю другу, тоді транспонувати матрицю.

---

18. Для прямокутної матриці знайти мінімальний з додатних елементів і максимальний з від'ємних елементів, після чого обміняти їх місцями. Нульові елементи не враховувати. Врахувати випадок, коли в матриці будуть відсутні додатні або від'ємні елементи.

---

19. В прямокутній матриці визначити кількість стовбців, повністю складених з додатних елементів.

Вказівка. Послідовний перегляд елементів стовбця припинити, якщо буде знайдений від'ємний або нульовий елемент.

20. Розглядаючи елементи рядка прямокутної матриці як координати точки в *n*-вимірному просторі, визначити номери точок, відстань між якими максимальна, після чого обміняти їх місцями.

---

21. Задана квадратна цілочисельна матриця. Якщо в її трикутній частини, розташованій нижче головної діагоналі, є непарні елементи, то замінити кожний з них середнім арифметичним значенням парних елементів стовбця, в якому розміщений непарний елемент. Середнє арифметичне значення округлити до найближчого цілого парного значення. Якщо в стовбці немає парних елементів, заміну не проводити.

---

22. Для заданої цілочисельної квадратної матриці розміром $n \times n$ елементів перевірити, чи співпадають *k*-ий рядок і *k*-ий стовбець (*k* = 1 .. *n*). Надрукувати номери співпадаючих рядків і стовбців.

---

23. Задана цілочисельна квадратна матриця. Згрупувати елементи кожного рядка, розташованого вище головної діагоналі, в порядку зменшення їхніх абсолютних значень.

---

24. Нульові елементи кожного стовбця прямокутної матриці зсунути на початок цього ж стовбця, зберігши без зміни вхідну послідовність інших елементів стовбця.

---

25. В кожному рядку квадратної матриці розмістити елементи в порядку зменшення їхніх абсолютних значень, не порушуючи при цьому положення елементів головної діагоналі.

---

26. В кожному рядку прямокутної матриці вилучити максимальний елемент, зсунувши на одну позицію ліворуч розташовані після нього елементи даного рядка. Останньому елементу рядка присвоїти нульове значення.

---

27. В прямокутній матриці визначити три мінімальні елементи і переставити їх в зворотному порядку.

Вказівка. В програмі виконати одноразовий перегляд елементів матриці.

28. В кожному рядку прямокутної матриці визначити різницю *d* поміж середнім арифметичним значенням *S*1 елементів, що розташовані на парних місцях, і середнім арифметичним значенням *S*2 елементів, що розташовані на непарних місцях. При цьому врахувати, що кількість стовбців може бути як парна, так і непарна. Згрупувати рядки в порядку зменшення параметру *d*.

---

29. Кожний рядок прямокутної матриці визначає координати точки в *n*-вимірному просторі. Точки належать до ломаної лінії, початок і кінець якої визначаються точками, що містяться в першому і в останньому рядках матриці. Обчислити довжину ломаної і визначити номер її відрізка, що має найбільшу довжину.

---

30. Для кожного рядка прямокутної матриці визначити кількість порушень *k* умови упорядкованості її елементів по зростанню, тобто умови *a*[*i*,*j*] <= *a*[*i*,*j*+1]. Згрупувати рядки в порядку зростання параметру *k*.

---

31. Розглядаючи в квадратній матриці діагональ, що з'єднує лівий нижній елемент з правим верхнім елементом, визначити мінімальний по модулю елемент, розташований вище даної діагоналі, і максимальний по модулю елемент, розташований нижче цієї ж діагоналі, після чого обміняти їх місцями.

---

32. В кожному стовбці прямокутної матриці перенести максимальний по модулю елемент в останню позицію стовбця, зсунувши при цьому вверх розташовані після нього елементи. Врахувати випадок, коли максимальний елемент вже знаходиться в останній позиції стовбця.

Приклад. Стовбець 5 -18 21 -12 10 -24 13 17 -8 10
після перетворення буде мати вигляд
5 -18 21 -12 10 13 17 -8 10 -24.

---

33. В прямокутній матриці визначити кількість стовбців, які складаються тільки з елементів одного знаку (додатних або від'ємних) і не містять нульових елементів. Перемістити такі стовбці в початкову частину матриці, зберігши їхній вхідний відносний порядок.

---

34. Для прямокутної матриці визначити значення і місцеположення елементу, що є сідловою точкою матриці (якщо така є).

Вказівка. Сідловою точкою матриці вважати не розташований на її периметрі елемент з індексами (*i*,*j*), що є мінімальним в *i*-ому рядку і водночас максимальним в *j*-ому стовбці.

---

35. В кожному стовбці прямокутної матриці визначити значення *Dmax* і номер *Kmax* елементу, який найбільше відрізняється від середнього арифметичного значення елементів даного стовбця, після чого згрупувати стовбці в порядку зменшення параметру *Dmax*.

---

36. Задана квадратна матриця, що складається з дійсних елементів. Згрупувати елементи кожного стовбця, розташовані нижче головної діагоналі, в порядку зростання їхніх абсолютних значень.

---

37. В кожному рядку прямокутної цілочисельної матриці визначити відсоток парних додатних чисел по відношенню до загальної кількості додатних чисел в даному рядку, після чого згрупувати рядки в порядку зменшення означеного відсотка.

---

38. Для кожного стовбця прямокутної матриці знайти різницю *d* поміж максимальним і мінімальним елементами, після чого згрупувати стовбці в порядку зменшення параметру *d*.

---

39. Задана квадратна цілочисельна матриця. Якщо в її трикутній частині, розташованій вище головної діагоналі, є непарні елементи, то замінити кожний з них середнім арифметичним значенням парних елементів рядка, в якому розміщений непарний елемент. Середнє арифметичне значення округлити до найближчого цілого парного значення. Якщо в рядку немає парних елементів, заміну не проводити.

---

40. Для прямокутної матриці сформувати одновимірний масив *B*, *j*-ий елемент якого дорівнює номеру першого нульового елемента в *j*-ому стовбці матриці, після чого згрупувати стовбці в порядку зменшення елементів *b*[*j*].

---

41. В прямокутній матриці визначити кількість рядків, які повністю складаються з від'ємних елементів, після чого перемістити ці рядки на початок матриці, зберігши їхній вхідний відносний порядок.

Вказівка. Послідовний перегляд елементів рядка припинити, якщо в ньому

буде знайдено додатний або нульовий елементи.

---

42. Задана квадратна цілочисельна матриця. Якщо в її трикутній частині, розташованій вище побічної діагоналі, є парні елементи, то замінити кожний з них середнім арифметичним значенням непарних елементів стовбця, в якому розміщений парний елемент. Середнє арифметичне значення округлити до найближчого цілого непарного значення. Якщо в стовбці немає непарних елементів, заміну не проводити.

---

43. Якщо максимальний елемент *i*-ого рядка прямокутної матриці (*i* = 1 .. *m*) більший за суму інших елементів цього рядка, а модуль мінімального елементу менший за суму інших елементів рядка, то замінити максимальний і мінімальний елементи півсумою їхніх значень, інакше рядок матриці залишити без змін. Підрахувати кількість рядків матриці, в яких проведена означена заміна.

---

44. В квадратній матриці всі елементи, розташовані нижче головної діагоналі, дорівнюють нулю. В кожному стовбці, окрім останнього, присвоїти елементу, розташованому безпосередньо нижче головної діагоналі, таке значення, щоб сума елементів в стовбці без врахування елементу головної діагоналі дорівнювала нулю.

---

45. Для прямокутної матриці *A* сформувати одновимірний масив *B, i*-ий елемент якого дорівнює номеру першого від'ємного елемента в *i*-ому рядку. Згрупувати рядки матриці в порядку зменшення елементів *b*[*i*].

---

46. Кожний стовбець прямокутної матриці визначає координати точки в *m*-вимірному просторі. Визначити номери точок, які найменше віддалені одна від одної, після чого обміняти їх місцями.

---

47. Задана прямокутна матриця цілих додатних чисел. Елементи кожного рядка матриці згрупувати в порядку зменшення і визначити після цього, для скількох елементів виконується умова *a*[*i,j*] < *j*.

---

48. Серед діагоналей квадратної матриці, які паралельні головній діагоналі і розташовані вище неї, знайти таку, в якій сума модулів елементів мінімальна у порівнянні з іншими діагоналями.

49. Кожний рядок прямокутної матриці містить координати точки в *n*-вимірному просторі. Визначити, скільки точок розміщено всередині гіперсфери з координатами центру (*c*1, *c*2,..., *cn*) і радіусом *R*, поза сферою і на її поверхні. Параметри *R* і *c*1,..., *cn* вводити з окремого файлу.

Примітка. Точка знаходиться всередині гіперсфери, якщо відстань від неї до центру гіперсфери менше за радіус гіперсфери.

---

50. В квадратній матриці знайти значення і місцеположення максимального елементу серед елементів, розташованих вище головної діагоналі, і мінімального елементу серед елементів, розташованих нижче головної діагоналі, після чого обміняти ці елементи місцями.

З А В Д А Н Н Я № 2

ВИКОРИСТАННЯ ПРОЦЕДУР І ФУНКЦІЙ

Методичні вказівки

Мета роботи – практично освоїти засоби організації процедур і функцій, а також засоби їхнього використання при рішенні задач на мові Паскаль. В кожному з наведених нижче завдань необхідно скласти програму, яка включає в свій склад принаймні одну підпрограму (процедуру або функцію), що виконує основну обробку вхідних даних в відповідності з умовою задачі. **Всі завдання сформульовані стосовно до обробки одного одновимірного масиву або пари масивів, що обробляються водночас. В програмі, що реалізує завдання №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 припускається, що масиви (або пари масивів), що послідовно в ній обробляються, мають різні імена типів.

Завдання №2 рекомендується виконувати в три етапи. В звіті по контрольній роботі потрібно навести лише матеріали останнього етапу.

Етап 1. Програма складається в припущенні, що обробляється тільки один масив (або одна пара масивів). На цьому етапі повинна бути ретельно перевірена правильність реалізації алгоритму рішення задачі шляхом прогонки системи тестових вхідних даних. В систему тестів повинні бути включені не тільки "нормальні" масиви, але і масиви, яким притаманні деякі граничні властивості

(наприклад, масив, що складається з одніх нулів, або масив, що складається з одного елементу). Для реалізації однотипних фрагментів програми рекомендується застосовувати процедури або функції; в складі програми доцільно застосовувати також інструментальні процедури і функції, що можуть бути використані практично без зміни в різноманітних програмах (наприклад, друк одновимірного масиву, контроль розміру сторінки, що друкується на екрані і т. і.).

Етап 2. Розроблена програма перетворюється в процедуру або функцію в припущенні, що всі масиви (або пари масивів), що послідовно обробляються, мають однакові імена типів. В основній програмі в цьому випадку виконується введення і друк вхідних даних, послідовне звернення до підпрограми, друк результатів. Кожний з масивів, що обробляються, повинен вводитися з окремого текстового файлу.

Етап 3. Виконується перетворення основної програми і підпрограм для загального випадку, коли передбачається обробка масивів, що мають різні імена типів.

Нижче на конкретному прикладі приводиться ілюстрація етапів виконання завдання №2.

Приклад.

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 задачі: з використанням і без використання буферних масивів.

В програмі Task2_2a використовуються два буферні масиви: в масив *Xpos* по мірі перегляду вхідного масиву *X* записуються додатні елементи, в масив *Xneg* - від'ємні. Кількість елементів в масиві *Xpos* дорівнює *j*, в масиві *Xneg* - *k*. Після цього в масив *X* переписується вміст масиву *Xpos*, потім дописуються нульові елементи, кількість яких дорівнює $m = n - j - k$, і на останньому етапі переписується масив *Xneg*.

```
Program Task2_2a;
Const Nmax=500;
Type Ar=array [1.. Nmax] of real;
Vari,                       { 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
```

```
      В 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 е д е н н я   X
End.
```

Другий варіант задачі реалізований в програмі Task2_2b. У зв'язку з забороною використання буферних масивів алгоритм її рішення істотно відрізняється від алгоритму попереднього варіанту.

На першому етапі роботи програми Task2_2b здійснюється перестановка додатних елементів на початок масиву *X* із збереженням їхнього вхідного відносного порядку. Перегляд елементів масиву *X* виконується в циклі **While**, починаючи з елементу з індексом *i* = 1. Якщо *i*-ий елемент додатний, то здійснюється перехід до наступного елементу *x*[*i*+1]; інакше за допомогою функції SearchePos виконується пошук найближчого додатого елементу *x*[*j*], після чого здійснюється зсув підмасиву *x*[*i*].. *x*[*j*-1] праворуч на одну позицію, а елементу *x*[*i*] надається вхідне значення елемента *x*[*j*]. Цикл **While** працює під керуванням булевої змінної CondPos і припиняє свою роботу в двох випадках:

- в циклі проаналізований останній елемент з індексом *i* = *n*;
- починаючи з індексу *i*+1, в масиві не виявлено додатних елементів.

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

При *i* <= *n* з підмасиву *x*[*i*+1] .. *x*[*n*] усуваються нульові елементи, а їхня кількість формується в лічильнику CountZero. Елементи зміненого масиву, починаючи з позиції *i*+1, зсуваються на CountZero позицій праворуч, а в "визволену" частину масиву дописуються нулі.

```
Program Task2_2b;
Label  10;
Const  Nmax = 500;
Type  Ar = array[1..Nmax] of integer;
Var  i,j,k,             { параметри циклів }
     n,                 { кількість элементів масиву X }
     CountZero : word;  { лічильник нульових элементів }
     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
       В 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 { підрахування кількості }
        If x[i]=0 then     { нульових елементів та }
          Begin            { вилучення їх із 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  {Зсув
підмасиву}
x[j]:=x[j-CountZero];
For j:=i to i+CountZero-1 do  { Записування нулів}
x[j]:=0;
End;
End;
      В и в е д е н н я   X
End.
```

В програмах Task2_2a і Task2_2b для скорочення їхнього тексту інструментальні процедури не наведені.

Етап 2. Перетворення частини програми Task2_2b, що реалізує алгоритм обробки даних, в підпрограму Transpos проілюстроване в наведеній нижче програмі Task2_3. При цьому припускаємо, що всі масиви, що обробляються, мають одне і те саме ім'я типу. Тут треба зауважити на таке.

1. Кожний з масивів *X, Y, Z*, що обробляються, розміщений в окремому файлі.
2. Введення і друк масивів, що обробляються, відбувається в основній програмі шляхом звернення до процедур ReadArray, ScreenArray і PrinterArray.
3. Всі інструментальні підпрограми незалежні від процедури Transpos і є внутрішніми тільки по відношенню до основної програми.
4. При обробці кожного з масивів *X, Y, Z* по описаному вище алгоритму відзнака складається лише в імені і розмірі масиву; в зв'язку з цим для процедури Transpos формальними параметрами взяті ім'я масиву *A*, що обробляється, і його розмір *n*.

5. Описи змінних, що наведені в розділі **Var** програми Task2_2b, поділені на глобальні (розділ **Var** програми Task2_3) і локальні (розділ **Var** процедури Transpos). До локальних змінних віднесені ті змінні, що використовуються тільки в процедурі переставлення елементів масиву, що обробляється.

```
Program Task2_3;
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, доки 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 із 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);
{ Виведення 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);
{ Виведення 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);
{ Переставлення елементів одновимірного масиву }
Var  i,j,k,              { параметри циклів }
    CountZero : word;    { лічильник нульових элементів }
    R : integer;         { буферна змінна }
    CondPos : boolean;   { змінна керування циклом }
Begin
  i:=1;
  CondPos:=true;
  While CondPos do       { Переставлення 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     { Підрахування кількості }
        If a[i]=0 then         { нульових елементів і }
          Begin                { вилучення їх із  }
            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 { Зсув підмасиву }
              a[j]:=a[j-CountZero];
            For j:=i to i+CountZero-1 do { Записування нулів}
              a[j]:=0;
          End;
      End;
End { Transpos };
{ ------------------------------------------------- }
Begin
{ Встановлення відповідності поміж внутрішніми }
{ та 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 будете використовувати принтер (Так,Ні) ?');
  ch:=ReadKey;
  If  ch in ['T','т','N','n'] 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('Перетв'orений 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 цьому етапі здійснюється доопрацювання програми Task2_3 в припущенні, що масиви *X, Y, Z*, що обробляються, мають різні імена типів.

Практично для кожної мови програмування створюються пакети прикладних математичних програм (ППП), що реалізують процедури чисельного аналізу, які часто зустрічаються на практиці (рішення нелінійних рівнянь, знаходження коренів поліномів, обертання матриці, рішення системи диференційних рівнянь і т. і.). Прикладна програма розробляється у вигляді підпрограми, звернення до якої відбувається з програми користувача. При цьому, як правило, ППП поставляються у вигляді об’єктних модулів, що виключає можливість зміни їхнього тексту користувачем.

Паскаль-підпрограма, що обробляє масив, повинна містити в списку формальних параметрів ім'я цього масиву з вказівкою відповідного імені типу. В загальному випадку ім'я типу формального масиву не співпадає з ім'ям типу фактичного масиву в програмі користувача. Отже, прикладна підпрограма повинна забезпечувати сумісність типів формального і фактичного масивів.

В Паскаль-програмі масив завжди має фіксований розмір, що визначається його ім'ям типу. Необхідність використання різних імен типів пов'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*, що мають різні імена типів, необхідно обчислити середнє арифметичне їхніх елементів. Тоді програма може мати ось такий вигляд.

```
Program Example;
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 A: Xar; Var S: real; n: word);
Var i : word;
Begin
  S:=0;
  For i:=1 to n do
    S:=S+a [i];
  S:=S/n;
End { MiddleAr };
{ ------------------------------------------------- }
Begin
  Введення 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 вірно сприйняте лише перше звернення до процедури MiddleAr; для інших звернень буде видане повідомлення про невідповідність типів фактичних параметрів *Y*, *Z* і формального параметру *A*.

Можливість обробки масивів з різними іменами типів може бути забезпечена одним із двох засобів: шляхом використання апарату приведення типів змінних або за допомогою абсолютних змінних. Як правило, для цього застосовується другий засіб. По відношенню до програми Example це показано в нижченаведеній програмі Example1.

```
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 A; Var S:real; n:word);
```

```
Type  RealAr = array[1..10000] of real;
Var    i : word;
   B : RealAr absolute A;
Begin
  S:=0;
  For i:=1 to n do
    S:=S+b[i];
  S:=S/n;
End { MiddleAr };
{ ------------------------------------------------------ }
Begin
  Введення 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.
```

В Турбо-Паскалі допускаються формальні параметри-змінні без типу. Таким формальним параметрам можуть відповідати фактичні параметри будь-якого типу. Це дозволяє передавати на обробку в процедуру масиви, проголошені в розділі **Var** з різними іменами типів.

В процедурі MiddleAr формальний параметр *A* - це параметр-змінна, оскільки перед ім'ям *A* стоїть слово **Var.** Це означає, що у разі звертання до процедури фіктивна адреса змінної *A* заміщується реальною адресою змінної *X, Y* або *Z*.

Як відомо, опис процедури передує розділу операторів основної програми. Компіляція програми здійснюється послідовно по мірі зчитування компілятором її тексту. Отже, в момент трансляції процедури компілятору не відомо, які фактичні параметри будуть відповідати формальним параметрам у разі звертання до процедури. Оскільки формальний параметр *A* проголошений без типу, то компілятор не має інформації про те, що означає ідентифікатор *A* - просту змінну, масив, запис і т. і. Додаткову інформацію з цього питання надає фраза

```
B: RealAr absolute A;
```

Ця фраза означає, що ідентифікатор *B* - це змінна типу RealAr, себто одновимірний масив з елементами типу real, при цьому адреса змінної *B* співпадає з адресою формальної змінної *A*. Отже, при зверненні MiddleAr(X,Sx,nx) змінні *X, A* і *B* мають один і той же адрес, себто визначають одне й те ж поле пам'яті. Оскільки в тілі процедури замість елементу *a*[*i*] записаний елемент *b*[*i*], то компілятор має достатню інформацію для перетворення операторів Паскаль-програми в машинні команди.

Тип RealAr, що використовувався для опису локальної змінної *B*, "довше" типів *Xar, Yar, Zar*. Однак при роботі процедури MiddleAr виходу за межі реальних масивів *X, Y, Z* не відбудеться, якщо змінні *nx, ny, nz* не перевищують відповідно значень 50, 500, 5000.

Адреса поля пам'яті - це адреса його крайнього лівого байта. Ця адреса не визначає ані структуру, ані довжину поля пам'яті. Така інформація міститься лише в описі типу відповідної змінної. По відношенню до локальної змінної *B* тип RealAr вказує, що *B* - це одновимірний масив, який має 10000 дійсних елементів. Якщо б після імені типу не було записано абсолютне речення, то змінній *B* при старті процедури було б виділено 10000*6=60000 байт пам'яті. Насправді цього не відбувається, оскільки фраза "**absolute** *A*" вказує, що змінній *B* повинна бути присвоєна та ж адреса, яку одержує формальна змінна *A*. У цьому контексті оголошення типу

```
RealAr = array [1..1] of real;
RealAr = array [1..10000] of real;
RealAr = array [1..2*MaxInt div SizeOf(real)] of real;
```

еквівалентні, оскільки вони інформують компілятор лише про те, що змінна типу RealAr - це одновимірний масив з дійсними компонентами. Останній варіант оголошення RealAr визначає максимально можливий розмір масиву (64 Кбайта), кількість його елементів 2*32767 **div** 6=10922.

```
Program Task2_4;
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 }
                          { принтеру }
    ch : char;            { символ натиснутої клавіши }
    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 до тих пір, доки 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 одновимірного масиву із 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);
{ Виведення на екран одновимірного 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);
{ Виведення на принтер одновимірного 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 додатного 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);
{ Переставлення елементів одновимірного масиву }
Var   i,j,k,            { параметри циклів }
CountZero : word;       { лічильник нульових элементів }
R : integer;            { буферна змінна }
CondPos : boolean;      { змінна керування циклом }
B : IntAr absolute A;
Begin
  i:=1;
  CondPos:=true;
  While CondPos do       { Переставлення додатних }
    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  { Підрахування кількості }
        If  b[i]=0 then     { нульових елементів і }
          Begin             { вилучення 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 { Зсув підмасиву }
            b[j]:=b[j-CountZero];
          For j:=i to i+CountZero-1 do   { Запис нулів }
            b[j]:=0;
        End;
    End;
End { Transpos };
{ ------------------------------------------------- }
Begin
{ Встановлення 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 принтеру }
  ClrScr;
  Writeln('Чи буде використаний принтер (Так,Ні) ?');
```

```
  ch:=ReadKey;
  If  ch in ['T','т','N','n'] then
    IndPrinter:=true
  Else
    IndPrinter:=false;

{ Обробка масиву X }
  ClrScr;
  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);
  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
```

На завершення розглянемо питання про використання буферних масивів в процедурах, що призначені для обробки масивів з різними іменами типів.

Цілком очевидно, що програма Task2_2a простіша у порівнянні з програмою Task2_2b. Водночас програма Task2_2a більш ефективна по швидкодії, оскільки в ній не відбуваються численні зсуви 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 означеної програми в процедуру, призначену для обробки масивів з різними іменами типів.

Кількість елементів в кожному з масивів Xpos і Xneg повинна бути такою самою, як і 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 яка реалізує алгоритм програми

Task2_2a, проголосити 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 суміщається за адресою з оброблюємим масивом X, Y або Z. Для буферних масивів вимагаються окремі поля пам'яті. В цьому випадку довелось би вказати для них тип максимального з оброблюємих масивів, що не сприяло б, зокрема, універсальності розробленої процедури. Перенесення оголошення буферних масивів в глобальний розділ опису змінних не покращило б ситуацію з неефективним використанням пам'яті. В цьому випадку прийшлося б проголосити або три пари буферних масивів відповідно з типами ArX, ArY і ArZ, або, як і при локальному описі, проголосити одну пару з максимальним типом ArZ.

У зв'язку з труднощами, що виникають при використанні буферних масивів в процедурах обробки масивів з різними іменами типів, в загальних зауваженнях до варіантів завдання №2 підкреслюється, що при їхній програмній реалізації буферні масиви не повинні застосовуватися.

Примітка. Ефективність програми Task2_2a при переході до процедур, що обробляють масиви з різними іменами типів, можна зберегти, якщо замість масивів Xpos і Xneg використати лінійні списки даних, в даному випадку - черги.

ВАРІАНТИ ЗАВДАННЯ № 2

Загальні зауваження

а) Якщо в умові задачі задана обробка одного масиву (наприклад, масиву *X*), то програма повинна обробляти три різних масиви (наприклад, *X*, *Y* і *Z*), причому в розділі **Var** ці масиви повинні мати різні імена типу (наприклад, **Var** `X: ArX; Y: ArY; Z: ArZ`).

б) Якщо в умові задачі задана обробка однієї пари масивів (наприклад, масивів *X* і *Y*), то програма повинна обробляти три різні пари масивів (наприклад, *X1* і *Y1*, *X2* і *Y2*, *X3* і *Y3*), причому в розділі **Var** ці пари масивів повинні мати різні імена типу (наприклад, **Var** `X1,Y1: Ar1; X2,Y2: Ar2; X3,Y3: Ar3`).

в) При розробці програми рішення задачі буферні масиви не використовувати.

---

1. Заданий масив цілих чисел *X* = (*x1*, *x2*,..., *xn*). Сформувати масив *Y*=(*y1*, *y2*,..., *ym*), помістивши до нього в порядку зменшення всі різні числа, що входять в масив *X*. Визначити, наскільки відрізняються середні арифметичні значення елементів масивів *X* і *Y*.

Наприклад, для масиву *X* = (5,8,3,5,8,7,5,8,4,1,4)

отримаємо $Y = (8, 7, 5, 4, 3, 1)$.

---

2. Задані два масиви $X = (x1, x2,.., xn)$ і $Y = (y1, y2,.., ym)$, до складу яких входять натуральні числа, причому в кожному з цих масивів немає елементів, що повторюються. Сформувати масив Z, включивши до нього всі елементи, що водночас містяться в масиві X і в масиві Y.

---

3. Заданий цілочисельний масив $X = (x1, x2,..., xn)$, в якому можуть бути однакові числа. Знайти максимальний і мінімальний елементи серед чисел, які не повторюються, і обміняти їх місцями. Врахувати окремі випадки, коли: в масиві немає чисел, які не повторюються; в масиві міститься лише одне число, яке не повторюється.

---

4. Із масиву цілих додатних чисел $X = (x1, x2,..., xn)$ усунути усі парні по значенню елементи, окрім останнього. Після цього числа, що залишилися, розташувати в порядку зростання. Врахувати окремі випадки, коли: в масиві немає парних елементів; є тільки один парний елемент; усі елементи парні. Буферний масив не використовувати.

---

5. В масиві $X = (x1, x2,..., xn)$ поміняти місцями перший і другий від'ємні елементи, третій і четвертий від'ємні елементи і т. д. Якщо кількість від'ємних елементів в масиві менше двох, перетворення масиву не проводити. Визначити, як змінилося положення мінімального і максимального елементів масиву X при його перетворенні.

---

6. Знайти максимальне і мінімальне з чисел, що зустрічаються в цілочисельному масиві $X = (x1, x2,..., xn)$ більше двох разів, і поміняти їх місцями. Перегляд масиву X виконати тільки один раз.

---

7. Відомо, що в цілочисельному масиві $X = (x1, x2,..., xn)$ три і тільки три числа рівні між собою. Знайти ці числа і розташувати їх на початку масиву, зсунувши інші числа до кінця цього масиву.

---

8. Перетворити масив X, розташувавши спочатку його від'ємні елементи, а потім усі елементи, які залишилися. При цьому в групі від'ємних елементів залишити їхній вхідний відносний порядок, а в групі елементів, що залишилася, змінити

його на зворотний. Визначити, як при цьому змінилося положення мінімального по модулю елементу масиву *X*.

---

9. Усунути із масиву дійсних чисел *X* = (*x1, x2,..., xn*) всі елементи, що перевищують його середнє арифметичне значення *S*, окрім першого такого елементу, і визначити, як при цьому змінилося значення *S*.

---

10. Всі додатні числа в масиві дійсних чисел *X* = (*x1, x2, ..., xn*) розташувати у зворотному порядку, не змінюючи при цьому положення інших чисел.

---

11. Для заданого значення *n* сформувати масив *X* = (*x1, x2,..., xn*), що є послідовністю чисел Фібоначчі. Оцінити, наскільки відрізняється положення елементу *x*[*i*], найбільш близького до середнього арифметичного значення масиву *X*, від положення елементу *x*[*j*], найбільш близького до середнього геометричного значення цього ж масиву.

Примітка. Послідовність чисел Фібоначчі формується по правилу:

$$f_{n+2} = f_{n+1} + f_n; \quad f_1 = 1; \quad f_2 = 1$$

Приклад: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,...

---

12. Заданий масив *X* = (*x1, x2,..., xn*), елементами якого є натуральні числа. В складі масиву *X* можуть бути елементи, що повторюються. Сформувати масив *Y*, включивши до нього всі елементи з масиву *X*, які не повторюються.

---

13. Задані два масиви *X* = (*x1, x2,..., xn*) і *Y* = (*y1, y2,..., ym*), до складу яких входять натуральні числа, причому в кожному з цих масивів немає елементів, що повторюються. Сформувати масив *Z*, об'єднавши масиви *X* і *Y*; при цьому в масиві *Z* також не повинно бути елементів, що повторюються.

---

14. В масиві *X* = (*x1, x2,..., xn*) розташувати в порядку зменшення додатні елементи, що входять до його складу, а після цього в порядку зростання – від'ємні елементи. Нульові елементи, якщо вони є в масиві *X*, розташувати поміж групою додатних і групою від'ємних елементів. Визначити, як при цьому змінилося положення максимального і мінімального елементів масиву X.

---

15. Переглянувши масив *X* = (*x1, x2,..., xn*) один раз, визначити значення і

положення (індекс) чотирьох найменших елементів, після чого розташувати їх в зворотному порядку.

---

16. Із масиву цілих додатних чисел $X = (x1, x2,..., xn)$ усунути всі непарні елементи, окрім першого такого елементу, після чого числа, що залишилися, розташувати в порядку зменшення.

---

17. Числова послідовність формується по правилу:

$$u0 = \cos(x);\ u1 = \cos(x+h);\ u2 = \cos(x+2h);\ ...;\ un = \cos(x+nh)$$

(значення x, n, h задані).

Серед тих елементів послідовності $U = (u0, u1,..., un)$, що перевищують по модулю задане значення b, знайти максимальний і мінімальний елементи, після чого обміняти знайдені елементи місцями. Врахувати, що в окремому випадку таких елементів може бути менш двох.

---

18. Задані два цілочисельних масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., ym)$. 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.

---

19. На початку заданого масиву $X = (x1, x2,..., xn)$ розташувати нульові елементи, що входять до його складу, а після цього в порядку зростання додатні елементи і в порядку зменшення від'ємні елементи. Визначити, як при цьому змінилося положення максимального і мінімального елементів масиву X.

---

20. Усунути із цілочисельного масиву $X = (x1, x2,..., xn)$ нульові елементи, розташувавши у зворотному порядку елементи, що залишилися. Визначити, як змінилися при цьому значення і положення мінімального і максимального елементів даного масиву.

---

21. Заданий цілочисельний масив $X = (x1, x2,..., xn)$, в якому можуть бути однакові числа. Знайти найближчу від початку масиву пару однакових чисел, різниця індексів між якими мінімальна.

---

22. Задані два масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., ym)$, до складу яких входять натуральні числа, причому в кожному з цих масивів немає елементів, що

повторюються. Сформувати масив *Z*, включивши до нього ті елементи із масиву *X*, що відсутні в масиві *Y*.

---

23. Переглянувши цілочисельний масив *X* = (*x1, x2,..., xn*) один раз, знайти два максимальних числа *Xmax1* і *Xmax2* відповідно серед парних і непарних по значенню елементів масиву, після чого розташувати в зворотному порядку елементи підмасиву, розташованого між цими числами, включаючи самі елементи *Xmax1* і *Xmax2*.

---

24. Всі парні числа в цілочисельному масиві *X* = (*x1, x2,..., xn*) розташувати в зворотному порядку, не змінюючи положення інших чисел.

---

25. Елементи масиву *X* = (*x1, x2,..., xn*) строго упорядковані по зростанню, тобто *x1* < *x2* < *x3* <... < *xn*, при цьому *x*[*i*+1] - *x*[*i*] > eps (*i*=1,..., *n*-1; де eps - мале число, наприклад, 0.001). Елементи масиву дійсних чисел *Y*= (*y1, y2,..., ym*), *m* <= *n*, розміщені у довільному порядку. Включити до складу масиву *X* ті елементи *y*[*j*], *j*=1.. *m*, що відрізняються від елементів *x*[*i*] не менше ніж на eps, зберігши при цьому упорядкованість масиву *X*.

---

26. Заданий масив *X* = (*x1, x2,..., xn*), елементами якого є натуральні числа. До складу масиву *X* можуть належати елементи, що повторюються. Сформувати масив *Y*, включивши до його складу по одному елементу, що повторюються в масиві *X*.

---

27. Заданий цілочисельний масив *X* = (*x1, x2,..., xn*), в якому можуть бути однакові числа. Знайти максимальний і мінімальний елементи серед чисел, що повторюються і обміняти їх місцями. Врахувати окремі випадки, коли: в масиві немає чисел, що повторюються; максимальний і мінімальний елементи дорівнюють один одному.

---

28. Виконати циклічний зсув масиву *X* = (*x1, x2,..., xn*) на *k* елементів (0 <= *k* <= *n*) у напрямку, що визначається значенням змінної *s* (*s* = 0 - ліворуч, *s* = 1 - праворуч). Наприклад, при циклічному зсуву масиву 4 -8 6 12 1 0 7 9 ліворуч на 3 елементи отримаємо 12 1 0 7 9 4 -8 6.
Значення змінних *k* і *s* ввести з клавіатури.

---

29. Задані два цілочисельних масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., ym)$, причому в кожному з них елементи можуть повторюватися. Вилучити з масивів X і Y всі елементи, що водночас містяться в обох масивах. Визначити, як при цьому змінилися максимальні і мінімальні значення елементів масиву X і масиву Y.

---

30. Із цілочисельного масиву $X = (x1, x2,..., xn)$ усунути всі непарні елементи, окрім останнього такого елементу.

---

31. Перевірити, чи є задана послідовність цілих додатних чисел перестановкою деякого відрізку ряду натуральних чисел. Наприклад, таким відрізком є послідовність

18 21 16 19 14 22 15 20 17

---

32. Елементи масивів $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., yn)$ визначають координати точок ламаної лінії. Усунути з складу ламаної відрізок мінімальної довжини і відрізок максимальної довжини. Визначити, як при цьому змінилася загальна довжина ламаної лінії і середня довжина її відрізків.

---

33. Перетворити масив X, розташувавши на його початку усі від'ємні елементи, а після них усі елементи, що залишилися. При цьому в групі останніх зберегти їхній вхідний відносний порядок, а в групі від'ємних елементів змінити його на зворотний. Визначити, як при цьому змінилося розташуваннння мінімального елементу масиву X.

---

34. Усунути із масиву $X = (x1, x2,..., xn)$ його максимальний і мінімальний елементи, після чого розташувать елементи, що залишилися, в зворотному порядку. Визначити, як змінилося середнє арифметичне значення елементів масиву X після його перетворення.

---

35. За одноразовий перегляд масиву $X = (x1,..., xn)$ визначити середнє арифметичне його трьох останніх додатних елементів. Врахувати окремий випадок, коли масив містить менше трьох таких елементів.

---

36. Цілочисельний масив $X = (x1, x2,..., xn)$ поділений на декілька підмасивів, при цьому для розподілу використовуються нульові елементи. Згрупувати елементи кожного підмасиву в порядку зростання. Врахувати окремі випадки, коли: в

масиві немає нульових елементів; підмасив порожній або містить лише один елемент.

---

37. В масиві $X = (x1, x2,..., xn)$ розташувати в порядку зростання додатні елементи, що входять до його складу, а після цього в порядку зменшення - від'ємні елементи. Нульові елементи, якщо вони є в масиві X, розташувати поміж групи додатних і групи від'ємних елементів. Визначити, як при цьому змінилося місцеположення максимального і мінімального елементів масиву X.

---

38. В масиві $X = (x1, x2,..., xn)$ поміняти місцями останній і передостанній додатні елементи, третій і четвертий від кінця масиву додатні елементи і т. д. Якщо кількість додатних елементів в масиві X менша за дві, перетворення масиву не проводити. Визначити, як змінилося положення мінімального і максимального по модулю елементів масиву X при його перетворенні.

---

39. Знайти мінімальне з чисел, що зустрічаються в цілочисельному масиві $X = (x1,..., xn)$ рівно два рази (якщо таке число є).

---

40. В масиві $X = (x1,..., xn)$ обміняти місцями максимальний елемент $Xmax$ з першим нульовим, а мінімальний елемент $Xmin$ - з останнім нульовим елементом масиву. Якщо в масиві тільки один нульовий елемент, то обміну підлягає той з елементів $Xmax$ і $Xmin$, що ближче розміщений до нульового елементу. Врахувати окремі випадки, коли: в масиві немає нульових елементів; максимальний або мінімальний елемент дорівнюють нулю.

---

41. Всі від'ємні числа в цілочисельному масиві $X = (x1, x2,..., xn)$ розташувати в зворотному порядку, не змінюючи положення інших чисел.

---

42. За одноразовий перегляд цілочисельного масиву $X = (x1, x2,..., xn)$ знайти його максимальний додатний елемент $Xmax$ і визначити середнє арифметичне значення всіх елементів масиву, за винятком елементів, рівних $Xmax$.

Вказівка. В програмі повинні бути враховані окремі випадки, коли: в масиві немає додатних елементів; всі елементи масиву додатні і дорівнюють одне одному.

---

43. За одноразовий перегляд цілочисельного масиву $X = (x1, x2,..., xn)$ знайти два

максимальних по модулю елемента, кратних відповідно числам 2 і 3, і, якщо такі елементи існують і вони не співпадають одне з другим, то поміняти їх в масиві місцями.

---

44. Задані масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., yn)$, елементи яких ($x[i]$, $y[i]$) являють собою координати точок на площині. Вважаючи кожні три суміжні точки (1,2,3), (2,3,4), (3,4, 5),... вершинами трикутника, визначити порядкові номери трикутників з максимальним і мінімальним периметрами, а також середнє значення периметрів усіх трикутників. Перегляд масивів виконувати тільки один раз.

---

45. Заданий масив цілих додатних чисел $X = (x1, x2,..., xn)$. Знайти в цьому масиві значення і положення максимального елементу, що є ступенем числа 2.

---

46. Задані масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., yn)$. 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),..., що лежать на одній прямій. Надрукувати номери першої і останньої груп таких точок (якщо вони є).

Примітка. Використати ось таку властивість: якщо три точки лежать на одній прямій, то площа трикутника з вершинами в цих точках дорівнює нулю. Подвоєну площу трикутника обчислювати за формулою:

$$S = x1*(y2-y3) + x2*(y3-y1) + x3*(y1-y2).$$

Площу трикутника вважати нульовою, якщо abs(S) < eps, де eps - мале число (наприклад, 0.001).

---

47. Масив дійсних чисел $X = (x1, x2,..., xn)$ містить декілька від'ємних елементів, що поділяють його на окремі підмасиви. Згрупувати елементи кожного підмасиву в порядку зростання. Врахувати окремі випадки, коли: в масиві немає від'ємних елементів; підмасив порожній, або містить тільки один елемент.

---

48. Визначити, чи містяться в заданій послідовності цілих додатних чисел $X = (x1, x2,..., xn)$ числа Фібоначчі і, якщо це так, визначити значення і місцеположення максимального і мінімального з таких чисел.

Вказівка. В програмі використати процедуру генерації чисел Фібоначі, що не перевищують значення елементу $x[i]$, який аналізується.

Правило формування чисел Фібоначчі дивись у завданні 11.

---

49. По заданому цілочисельному масиву $X = (x1, x2,..., xn)$ сформувати масив $Y = (y2,..., yn)$ такий, що $y[i]$ – це кількість елементів із X, які не перевершують $x[i]$ на початковому відрізку X від першого елементу до елементу з індексом i-1.

---

50. Задані два цілочисельних масиви $X = (x1, x2,..., xn)$ і $Y = (y1, y2,..., ym)$. Нехай в масиві X є $k1$ парних елементів, а в масиві Y - $k2$ непарних елементів. Обміняти місцями $k = \min(k1, k2)$ парних елементів масиву X з непарними елементами масиву Y (в їхній послідовності в масивах X і Y). Врахувати, що в окремому випадку може бути $k = 0$.
