Алгоритми на масивах мовою Pascal

Автор: Негода Сергій Петрович

Практична робота 25.

Алгоритми на масивах

Приклади кодування масивів мовою програмування Pascal.

1.ConstNamel: array[1..10]ofreal = (1, 3, 5, 7, 9, 0, 2, 4, 6, 8);

Сталий(постійний) одновимірний(лінійний) масив Namel із 10 дійсних чисел; діапазон зміни індексів 1..10.

2. var a: array [1..10] of real; begin for i:=1 to 10 do a[i]:=random(50); end.

Змінний(не постійний) одновимірний(лінійний) масив a із 10 дійсних випадкових чисел; діапазон зміни індексів 1..10.

3. const n=1; m=10; var Name2: array [n..m] of real;

Змінний одновимірний масив Name2 із 10 дійсних чисел; діапазон зміни індексів 1..10.

4. type Massiv=array [1..10] of real; var Name3: Massiv;const n=1; m=10;

5.const n=1; m=10; type Massiv = array [n..m] of real; var Name3: Massiv;

Тип Massiv як множина всіх одновимірних масивів із 10 дійсних чисел з діапазоном зміни індексів 1..10. Змінний одновимірний масив Name3 як проста змінна типу Massiv.

Завдання 1. *Ввести з клавіатури в масив 20 цілих чисел.*

```
program Zadacha1;

var A: array [1..10] of  integer; i: integer;

begin

  for i:=1 to 20 do read (A{[i]);

end.
```

Завдання 2. *Обчислити суму елементів масиву з n цілих чисел.*

```
program Zadacha2;
const n=10;
var a: array [1..n] of integer; i, sum: integer;
begin
  for i:=1 to n do
  begin
    write ('Введіть',i, '- й елемент масиву'); read (a[i]);
  end;
  sum:=0;
  for i:=1 to n do sum:=sum + a[i];
  writeln ('Сума всіх елементів масиву дорівнює: ', sum);
end.
```

Завдання 3. *Знайти у масиві місце розташування елементів із заданим значенням.*

```
program Zadacha3;
const n=4;
var a: array [1..n] of real;
  x: real; i: integer;
begin
  writeln (Введіть елементи масиву');
  for i:=1 to n do read (a[i]);
  writeln ('Введіть значення для пошуку');
  read (x);
  for i:=1 to n do
    if a[i]=x then
    writeln ('на',i, '- місці масиву розташований заданий елемент',x:8:2);
end.
```

Завдання 4. *(Бінарний пошук) Знайти заданий елемент у впорядкованому за зростанням масиві, якщо відомо, що він існує.*

```
program Zadacha4;

const n=4;

var a: array [1..n] of real; x: real; i, m, k, i: integer;

begin

  writeln ('Введіть елементи масиву');

  for i:=1 to n do read (a[i]);

  writeln ('Введіть значення для пошуку'); read (x);

  1:=1; m:=n; k:=n div 2;

  While a [k] < > x do

  Begin

    k:= (1+m) div 2;

    if a [k] > x then m:= k-1; if a[k] <  x then 1:=k+1;

  end;

  writeln ( 'заданий елемент', x:8:2, 'на',  k , '- му місці');

end.
```

Завдання 5. *Знайти найменший(мінімальний) елемент масиву.*

```
program Zadacha5;

const n=4;

var A : array [1..n] of real;

  min: real; i: integer;

begin

  writeln ('Введіть елементи масиву');

  for i:=1 to n do read (A[i]);

  min:= A [1];
```

```
  for i:=2 to n do
    if A[i] < min them min:= A[i];
  writeln ('мінімум масиву=' , min);
end.
```

Практична робота 26.

Алгоритми табуляції перестановок

Завдання 1. Скласти і реалізувати алгоритм для табулювання значень перестановок натуральних чисел. Наприклад: Перестановки для трьох чисел: 123 132 213 231 321 312(усього шість).

```
ProgramPerestanovka1;                    { оголошення назви алгоритму }
constn=5;                    { оголошення константи, кількість чисел в перестановці }
var a: array [1..n] of integer;    { оголошення масиву цілих чисел для запису перестановок }
    index: integer;              { оголошення цілої змінної для кодування чисел в перестановках}
procedure generate(l, r: integer);        { оголошення підпрограми генератора перестановок }
var i, v: integer;                        { оголошення змінних цілих чисел для алгоритму }
begin       if (l = r) then begin          { оголошення перевірки рівності двох індексів }
for i:=1 to n do write (a [i], ''); { оголошення циклу з лічильником для виведення перестановок }
      writeln;
    end   else   begin
      for i:=l to r do begin   { оголошення циклу з лічильником для генерації перестановок }
        v:=a[l]; a[l]:=a[i]; a[i]:=v;                {Обмін a[i], a[j]}
```

generate (l + 1, r); {Виклик нової генерації перестановок - це рекурсія в алгоритмі}

v:=a[l]; a[l]:=a[i]; a[i]:=v; {Обмін a [i], a [j]}

end; end; end;

begin

for index:=1 to N do a[index]:=index; { цикл з лічильником для генерації перестановок }

generate(1, n); end. {виклик процедури для генерації перестановок }

Протестуйте алгоритм чотири рази, тобто треба змінити число табуляції:

1)const=3; 2) const=4; 3) const=6.

Завдання 2. Є масив, що складається з N чисел. За один крок дозволяється зменшити на 1 кілька (можливо один) підряд рівних елементів. Мета – зробити всі елементи рівними нулю. За яку мінімальну кількість кроків це може бути зроблено?

Формат введення-виведення: Програма зчитує з клавіатури (стандартного пристрою введення) натуральне число N ($1 \le N \le 2 \cdot 10^5$) – кількість чисел у масиві, а з наступного рядка N невід'ємних цілих чисел, елементів масиву, кожне з яких не перевищує $2 \cdot 10^9$.
Програма виводить на екран (стандартний пристрій виведення) єдине число – шукану кількість кроків.

```
PROGRAm Zeroing1;

type arr= Array[1..200005] of longint;

var  A:arr;      i, k, n: longint;

begin   read(n);    for i:=2 to n+1 doread(a[i]); writeln(' ****  n= ', n);  a[1]:=0;   k:=0;

  for i:=1 to n do   begin     writeln('*** a[',i,']=', a[i]); writeln(' a[',i+1,']=', a[i+1]);

    if a[i+1]>a[i] then k:=k+a[i+1]-a[i];   writeln(' k=', k);   end;

write(' Найменша кількість кроків  для обнулення елементів масиву  k=', k);  end.
```

Дані для тестування алгоритму: А)Введення: 3; 3; 4; 1; Виведення результату : 4.

Б) Введення : 3; 3; 1; 4; Виведення: 6. В) Введення : 5; 4; 2; 5; 4; 4. Виведення: 7.

Практична робота 27.

Алгоритми на масивах

Завдання 1.Створити та реалізувати алгоритм мовою Pascal, який знаходить суми усіх

чисел в одномірному масиві(рядок чисел), якщо у масиві алгоритмом задаються випадкові трицифрові цілі додатні числа.

Наприклад: Дано масив чисел: (1; 3; 5; 8; 0; 3; 9). Його сума: 1+ 3+5+8+0+3+9=29.

programSUMMA1; {Підрахунок суми усіх елементів числового масиву}

```
const  n=20;    var a: array [1..n] of integer;        s,     i: integer;

begin       writeln ( 'Вводиться    ', n,  '  випадкових елементів масиву');

    for i: = 1 to n do begin     a[i]:=100+random(900);

writeln('випадковий  a[', i, ']-ий елементмасиву: ', a[i]);  writeln; end;

    s:=0;       for i:=1 to n do s:=s+a [i];

    writeln('Сума усіх елементів масиву =', s); writeln; end.
```

Протестувати алгоритм для 5-цифрових: а) const =5; б)const =15.

Завдання 2.Створити та реалізувати алгоритм мовою Pascal, який **знаходить середнє арифметичне усіх чисел** в двовимірному масиві(таблиця чисел), якщо у масиві алгоритмом задаються випадкові чотирицифрові цілі додатні числа.

programSUMMA2; {Середнє арифметичне елементів числового масиву}

```
const  n=3;       var  a: array [1..n, 1..n] of integer;          s, m, j, i: integer;

begin       writeln ( 'Вводиться    ', n*n,  '  випадкових елементів масиву');

    for    i:=1 to n do begin          for    m:=1 to n do begin

     a[i, m]:=1000+random(9000);

writeln('випадковий  a[', i, ' ,' , m, ' ]-ий елементмасиву: ', a[i, m]);  writeln; end; end;

    s:=0;   for i:=1 to n do begin       for   j:=1 to n do begin     s:=s + a[i, j];end;  end;
        writeln('Середнє арифметичнеелементів масиву =', s/(n*n)); writeln; end.
```

Протестувати алгоритм для 6-цифрових: а) const =2; б)const =4.

Завдання 3.Створити та реалізувати алгоритм мовою Pascal, який знаходить **найбільше та найменше** із усіх чисел в двовимірному масиві(таблиця чисел), якщо у масиві алгоритмом задаються випадкові 5-цифрові цілі додатні числа.

programMINMAX3; {Середнє арифметичне елементів числового масиву}

```
const  n=3;       var  a: array [1..n, 1..n] of integer;    s, m, max, min, j, i: integer;

begin       writeln ( 'Вводиться    ', n*n,  '  випадкових елементів масиву');

    for    i:=1 to n do begin          for    m:=1 to n do begin
```

a[i, m]:=10000+random(90000);

writeln('випадковий a[', i, ' ,' , m, ']-ий елементмасиву: ', a[i, m]); writeln; end; end;
s:=0; min:=a [1,1]; for i:=1 to n do begin for j:=1 to n do begin

if max <a [i, j] then max:= a [i,j]; if min>a [i, j] then min:=a[i,j]; end; end;

writeln('Мінімальний елемент масиву =', min); writeln;

writeln('Максимальний елемент масиву =', max); writeln;

writeln('Середнє арифметичне MAX та MIN =', (max+min) div 2); writeln; end.

Протестувати алгоритм для 7-цифрових: а) const =5; б)const =6.

Практична робота 28.

Алгоритми на масивах

Завдання 1.Створити та реалізувати алгоритм мовою Pascal, який знаходить кількість усіх чисел в одномірному масиві(рядок чисел), які **діляться на 3 націло**, якщо у масиві алгоритмом задаються випадкові шестицифрові цілі додатні числа.

Наприклад: Дано масив чисел: (1; 3; 5; 8; 0; 3; 9). Його сума: 1+ 3+5+8+0+3+9=29.

programSUMMA1; {Підрахунок суми усіх елементів числового масиву}

const n=20; var a: array [1..n] of integer; s, i: integer;

begin writeln ('Вводиться ', n, ' випадкових елементів масиву');

for i: = 1 to n do begin a[i]:=100+random(900);

writeln('випадковий a[', i, ']-ий елементмасиву: ', a[i]); writeln; end;

s:=0; for i:=1 to n do if (a[i] mod 3)=0 then s: =s+1;

writeln('Кількість елементів масиву, які діляться на 3: ', s); writeln; end.

Протестувати алгоритм для 5-цифрових: а) const =2; б)const =3; в) const =4 .

Завдання 2.Створити та реалізувати алгоритм мовою Pascal, який знаходить **кількість парних чисел головної діагоналі** в двовимірному масиві(таблиця чисел), якщо у масиві алгоритмом задаються випадкові семицифрові цілі додатні числа.

programPARA2; { Кількість парних чисел на діагоналі масиву}

const n=3; var a: array [1..n, 1..n] of integer; s, m, j, i: integer;

begin writeln ('Вводиться ', n*n, ' випадкових елементів масиву');

for i:=1 to n do begin for m:=1 to n do begin

a[i, m]:=100+random(9000);

writeln('випадковий a[', i, ' ,' , m, ']-ий елементмасиву: ', a[i, m]); writeln; end; end;

s:=0; for i:=1 to n do begin if (a[i, i] mod 2)=0 then s: =s+1; end;

writeln('Кількість парних чисел на діагоналі масиву =', s); writeln; end.

Протестувати алгоритм для 4-цифрових: а) const =2; б)const =4.

Завдання 3.Створити та реалізувати алгоритм мовою Pascal, який знаходить різницю найбільшого та найменшого із усіх чисел в двовимірному масиві(таблиця чисел), якщо у масиві алгоритмом задаються випадкові цілі від'ємні та додатні числа.

programDELTAMAXMIN3; { **Різниця MAX та MIN** числового масиву}

const n=3; var a: array [1..n, 1..n] of integer; s, m, max, min, j, i: integer;

begin writeln ('Вводиться ', n*n, ' випадкових елементів масиву');

for i:=1 to n do begin for m:=1 to n do begin

a[i, m]:=-(1000+random(90))* (random(9)-random(9)-2);

writeln('випадковий a[', i, ' ,' , m, ']-ий елементмасиву: ', a[i, m]); writeln; end; end;
s:=0; min:=a [1,1]; for i:=1 to n do begin for j:=1 to n do begin

if max <a [i, j] then max:=a[i,j]; if min>a [i, j] then min:=a[i,j]; end; end;

writeln(' Мінімальний елемент масиву =', min); writeln;

writeln(' Максимальний елемент масиву =', max); writeln;

writeln('Різниця MAX та MIN =', max-min); writeln; end.

Протестувати алгоритм для 3-цифрових: а) const =2; б)const =4.

Практична робота 29.

Обчислювальні алгоритми на масивах

Завдання 1.Створити та реалізувати алгоритм мовою Pascal, який підносить **до квадрату парні остачі усіх чисел в одномірному масиві(рядок чисел) при діленні на 7** і підносить **до кубу непарні остачі** (mod 7) та виводить результати цих дій у новий масив. І до того ж знаходить суму усіх цих остач. При цьому у початковому масивічисла задаються

алгоритмом, як випадкові 7-цифрові цілі від'ємні числа.

```
programMOD1;    {Підрахунок і дії з остачами для усіх елементів числового масиву}

const  n=2;    var a, b: array [1..n] of integer;        s, i: integer;

begin       writeln ( 'Виводиться    ', n,  '  випадкових елементів масиву A:');

 for   i:=1 to  n   do begin     a[i]:=-1000-random(9000);  write(' a[', i, ']=: ', a[i]);  write('
    '); end; writeln;  writeln ( 'Виводиться    ', n,  ' елементів масивуостач:');

  s:=0;       for i:=1 to n do  begin b[i]:=a[i]  mod  7;     s:=s+b[i];   write('b[', i, ']= ', b
[i]);  write('  '); end; writeln; writeln('Сумаусіх остач елементів масиву =', s);
writeln('***');

for i:=1 to n do  begin if  b[i] mod 2=0 then b[i]:= b[i]* b[i]  else b[i]:= b[i]* b[i]* b[i];

write('c[', i, ']= ', b[i]);  write('     '); end; writeln;  writeln('***********');end.
```

Протестувати алгоритм для 9-цифрових: а) const =9; б)const =8; в)const =25.

Завдання 2.Створити та реалізувати алгоритм мовою Pascal, який **знаходить парні числаі ділить ці числа на 2 та знаходить непарні числа і подвоює їх** в двовимірному масиві(таблиця чисел). При цьому у початковому масиві алгоритмом задаються випадкові 8-цифрові цілі, як додатні так і від'ємні числа.

```
programODD2;   {Дії окремо з парними  та окремо з непарними елементами масиву}

const  n=3;       var  a: array [1..n, 1..n] of integer;            j,i, p: integer;

begin       writeln ( 'Вводиться    ', n*n,  '  випадкових елементів масиву');

for  i:=1 to n do begin  for j:=1 to n do begin       a[i, j]:=-400+random(900);

write('  a[', i, ' ,' , j, ' ]= ', a[i, j]);  write('  '); end;    writeln('  ');  end;  writeln('Масив:');

    p:=0;   for i:=1 to n do begin       for   j:=1 to n do begin

  if  (a[i, j] mod 2)=0   then begin  p:=p + 1; a[i, j]:=a[i, j] div 2  end  else a[i, j]:=2*a[i, j];

write('  b[', i, ' ,' , j, ' ]=', a[i, j]);     write('  '); end;    writeln('  ');  end;  writeln;

writeln('Кількість парних елементів масиву =', p);

writeln('Кількість непарних елементів масиву =', n*n- p); writeln('*******');end.
```

Протестувати алгоритм для 4-цифрових: а) const =4; б)const =5; в)const =8.

Завдання 3.Створити та реалізувати алгоритм мовою Pascal, який виводить два масиви розміром nxn у вигляді рядків та стовпців і знаходить **потроєну суму та потроєну різницю цих двох** двовимірних масивів А та В(таблиці з різними числами, котрі збільшені в 3 рази), якщо у двох масивах алгоритмом задаються випадкові 6-цифрові цілі додатні числа.

programSUM3DELTA3; { Потроєна сума або потроєна різниця масивів: C= 3A ± 3B}

const n=2; var b, a: array [1..n, 1..n] of integer; k,m, j, i: integer;

begin writeln ('Виводиться по ', n*n, ' випадкових елементів масивів A та B');

for i:=1 to n do begin for j:=1 to n do begin a[i,j]:= 1000+random(900);

write(' a[',i,';',j,']:= ' , a[i,j]); end; writeln(' '); end; writeln(' Другий масив:');

for i:=1 to n do begin for j:=1 to n do begin b[i,j]:= 100000+random(90000);

write(' b[',i,';',j,']:= ', b[i,j]); end; writeln(' '); end; writeln(' ');writeln(' Відповідь.');

fori:=1 to n do beginfor j:=1 to n do begin write(' 3a+3b[', i, ' ,' , j, ']= ', 3*(a[i,j]+b[i,j])); end; writeln(' '); end; writeln(' '); writeln(' '); for k:=1 to n do begin for m :=1 to n do begin write(' 3a-3b[', k, ' ,' , m, ']= ', 3*(a[k,m]- b[k,m])); end; writeln(' '); end; writeln(' '); writeln; end.

Протестувати алгоритм для 10-цифрових: а) const =3; б)const =4; в)const =9.

Практична робота 30.

Обчислювальні алгоритми на масивах

Завдання 1.Створити та реалізувати алгоритм мовою Pascal, який знаходить потроєний **квадрат суми усіх чисел в одномірному масиві, тобто** $C= 3(a_1+a_2+...+a_p)^2$ і виводить новий масив, в якому кожний і-ий елемент дорівнює числу: $C - a_i * a_i$, якщо у даному масиві А алгоритмом задаються випадкові 7-цифрові цілі від’ємні числа.

programQUADRATSUMMA1; {Підрахунок потроєного квадрату суми чисел масиву}

const n=2; var a, b: array [1..n] of integer; c, s, i: integer;

begin writeln ('Вводиться ', n, ' випадкових елементів масиву A:');

for i:=1 to n do begin a[i]:=-100-random(900);

write(' a[', i, ']=: ', a[i]); write(' '); end; writeln;

writeln ('Виводиться ', n, ' елементів нового масивурізниць $C - a_I * a_I$**:');**

s:=0; for i:=1 to n do s:=s+a[i]; C:=3*s*s;

for i:=1 to n do begin write('b[', i, ']= ',c- a[i]*a[i]); write(' '); end; writeln;

writeln('Потроєний квадрат сумиелементів масиву A:', C); writeln('**************'); end.**

Протестувати алгоритм для 4-цифрових: а) const =7; б)const =6.

Завдання 2.Створити та реалізувати алгоритм мовою Pascal, який змінює усі числа на протилежні і обмінює числа місцями **відносно головної діагоналі** в двовимірному масиві, якщо у масиві алгоритмом задаються випадкові 8-цифрові цілі додатні і від'ємні числа.

```
programTRANSPONUVAN2;  {Обмін елементів масиву відносно головної діагоналі}

const  n=3;      var  a,b: array [1..n, 1..n] of integer;          s, m, j,i, p: integer;

begin      writeln ( 'Вводиться    ', n*n,  '  випадкових елементів масиву');

    for    i:=1 to n do begin          for    j:=1 to n do begin

     a[i, j]:=1000+random(9000)*(1- random(3));

write('  a[',i,';',j,']:= ' ,  a[i,j]);   end; writeln('   '); end;  writeln(' Змінений  масив:');

p:=0;   for i:=1 to n do begin       for   j:=1 to n do begin

 b[i, j] :=- a[j,i];  write('  b[', i, ' ,' , j, ' ]=', b[i, j]);  write('  '); end; writeln('  ');  end;
writeln;  writeln('*****************');end.
```

Протестувати алгоритм для 4-цифрових: а) const =2; б)const =4; в)const =5.

Завдання 3.Створити та реалізувати алгоритм мовою Pascal, який виводить два масиви розміром nxm у вигляді рядків та стовпців і **знаходить поелементу різницю між потроєними числами першого рядка масиву та подвоєними числами другого рядка** двовимірного масиву A(nxm) і результат записує у третій рядок, якщо у масиві A алгоритмом задаються випадкові 1-цифрові цілі додатні числа.

```
programDELTA3;   {Різниця між потроєними числами першого рядка масиву }

const  n=3; const  m=4;      var  b, a: array [1..n, 1..m] of integer;     j, i: integer;

begin   writeln ( 'Виведення ', n*m,  '  випадкових елементів масиву A: ');

for i:=1  to n do   begin      for j:=1  to m do  begin   a[i,j]:= 1+random(10) ;

 write('   a[',i,';',j,']:= ' ,  a[i,j]);   end; writeln('    ');  end;  writeln('   ');

writeln('Шуканий масив: ');

 for i:=1  to m do   begin  a[3, i]:= 3*a[1, i]- 2*a[2, i]; end;

 for i:=1  to n do   begin      for j:=1  to m do  begin

write('   b[',i,';',j,']:= ' ,  a[i,j]);   end; writeln('    ');  end;  writeln('   ');

writeln;   end.
```

Протестувати алгоритм для а) випадкових 10-цифрових: const n=6; const m=9;

б) випадкових 7-цифрових: const n=7; const m=8; в) 4-цифрових: const n=4; const m=5.

Практична робота 31.

Алгоритми на масивах

Завдання 1. Створити та реалізувати алгоритм мовою програмування Pascal, що містить масиви з дійсними випадковими числами і знаходить суму лише від’ємних чисел із масиву.

```
program summa1;

var   x: array[1..13] of real;     suma: real;   k, i: integer;

begin k:=0;

for i:=1 to 13 do   begin    x[i]:=50-random(100);

writeln('випадкове число в масиві  x[‘ , i, ‘]=’, x[i]);

  if x[i] < 0   then   begin k:=k+1;   suma:=suma+x[i];

 writeln('відємне число  x[‘ , i, ‘]=’, x[i]);    end; end;

writeln('**********');      writeln('сума відємних чисел в масиві = ',suma);

writeln(' кількість відємних чисел в масиві k= ', k); writeln('**********');  ;  end.
```

Завдання 2. Створити та реалізувати алгоритм мовою програмування Pascal, що відсортовує в порядку зростання лінійний масив з цілими випадковими числами в порядку зростання.

```
program  sort2;

var  a: array [0..101] of integer;

   kmin, b, i, j, k: integer;

begin     writeln(' Лінійний  масив   на випадкових числах');

for  i:=0 to 101 do begin  a[i]:=50-random(100);

writeln(' Випадкове число в масиві  a[' , i, ']=', a[i]);  end;

for i:=0 to 101 do   begin    kmin:= i;

    for j:=i+1 to 101   do   if   a[kmin]> a[j]   then    kmin:=j;

 b:=a[i];      a[i]:=a[kmin];     a[kmin]:=b;   end;

 writeln(' Відсортований  масиві  в порядку зростання');
```

for i:=0 to 101 do writeln(' a[' , i, ']=', a[i]); end.

Завдання 3. Створити та реалізувати алгоритм мовою програмування Pascal, що знаходить значення виразу $3x^2 + a$ з дійсними випадковими числами x в порядку зростання.

programtabuljacia3;

var f,x,d: real; n, i: integer; {оголошуються змінні типу: дійсні та цілі}

begin writeln('вводиться початкове випадкове число, яке є аргументом функції ');

x:=-(10+random(10)); write(x); writeln; n:= 10+random(20); d:= 1+random(5);

write('якщо кроків n =', n); writeln; write('якщо довжина кроку d =', d); writeln;

for i:=1 to n do begin {виконується цикл з лічильником для обчислення функції }

if x<-10 then f:=3*x*x+1 else {розгалуження для перевірки аргументу функції }

if x<10 then f:= 3*x*x+2 else f:= 3*x*x+3 ; x:=x+d; { розгалуження та обчислення}

writeln('результат обчислення f(', x, ')=', f, 'якщо номер кроку i =', i); writeln; end; end.

Завдання 4. Створити, реалізувати алгоритм мовою Pascal, який містить вибір ***case.***. Це нелінійний алгоритм, бо містить вибір на 6 випадків для цілих значень k від 0 до 5.

programtabul4;

var a, r, l, n, y, x: real; k: integer;

begin x:=random(80); writeln(' x1 :=',x); n:=random(90); writeln(' n1 :=',n);

a :=random(70); writeln(' a1 :=',a); y:=random(60); writeln(' y1 :=', y);

l:=random(60); writeln(' l1:=',l); k:=random(6); writeln(' k1 :=',k);

case k of

1: a:=-5*(20*sqrt(a) - 3.467/exp(a));

2: n:=-10/sqr(l)+790.89*exp(-n);

3: x:=1/sin(-n)+200*cos(y);

else begin y:=1; n:=1; x:=1; r:=1; a:=1; end;end;

writeln('r2:=', r); writeln('y2:=', y); writeln('k2:=', k); writeln('n2:=', n);

writeln('числоа2:=', a); writeln('числol2:=', l); writeln('********'); end.**

Практична робота 32.

Обчислювальні алгоритми на масивах

Завдання 1.Створити та реалізувати алгоритм мовою Pascal, який **знаходить суму чисел в кожному стовпчику** двовимірного масиву(таблиця чисел) і записує ці суми у одномірний масив. При цьому у початковому масиві nxn алгоритмом задаються випадкові дійсні, як додатні так і від'ємні числа.

programSUMMA1; {сума чисел у кожному стовпчику масиву}

```
const n=3; const m=4; var a: array [1..n, 1..m] of real; b: array [1..m] of real; j,i, p: integer;

begin writeln ( 'Вводиться ', n*n, ' випадкових елементів масиву');

for i:=1 to n do begin for j:=1 to m do begin a[i, j]:=-2-random(5)+0.8*random(4);

write(' a[', i, ' ,' , j, ' ]= ', a[i, j]); write(' '); end; writeln(' '); end;

writeln('Масив сум по стовпичках:'); for i:=1 to m do begin b[i]:=0; end;

for i:=1 to n do begin for j:=1 to m do b[j]:=a[i, j]+ b[j]; end;

for i:=1 to m do begin write(' b[', i, ' ]=', b[i]); write(' '); end; writeln;

writeln('*******');end.
```

Протестувати алгоритм для: а) const n=5; constm=6; б) constn=8; constm=8; в) const n=9 constm=10; г) const n=7; constm=10; д) constn=1; constm=10.

Завдання 2.Створити та реалізувати алгоритм мовою Pascal, який **знаходить суму чисел в кожному рядку** двовимірного масиву(таблиця чисел) і записує ці суми у одномірний масив. При цьому у початковому масиві nxn алгоритмом задаються випадкові дійсні, як додатні так і від'ємні числа.

programSUMMA2; {сума чисел у кожному рядку масиву}

```
const n=3; const m=4;

var a: array [1..n, 1..m] of real; b: array [1..n] of real; j,i: integer;

begin writeln ( 'Вводиться ', n*n, ' випадкових елементів масиву');

for i:=1 to n do begin for j:=1 to m do begin a[i, j]:=-2-random(5)+0.8*random(4);

write(' a[', i, ' ,' , j, ' ]= ', a[i, j]); write(' '); end; writeln(' '); end;

writeln('Масив сум по рядках:'); for i:=1 to n do begin b[i]:=0; end;

for i:=1 to m do begin for j:=1 to n do b[j]:=a[j, i]+ b[j]; end;
```

for i:=1 to n do begin write(' b[', i, ']=', b[i]); write(' '); end; writeln;

writeln('*****');end.**

Протестувати алгоритм для: а) const n=2; constm=6; б) constn=3; constm=8; в) const n=1 constm=10; г) const n=3; constm=5; д) constn=1; constm=1.

Завдання 3.Створити та реалізувати алгоритм мовою Pascal, який виводить два масиви розміром nxn у вигляді рядків та стовпців і знаходить **потроєну суму та потроєну різницю цих двох** двовимірних масивів А та В(таблиці з різними числами, котрі збільшені в 3 рази), якщо у двох масивах алгоритмом задаються випадкові 6-цифрові цілі додатні числа.

programSUM3DELTA3; { Потроєна сума або потроєна різниця масивів: С= 3А ± 3В}

const n=2; var b, a: array [1..n, 1..n] of integer; k,m, j, i: integer;

begin writeln ('Виводиться по ', n*n, ' випадкових елементів масивів А та В');

for i:=1 to n do begin for j:=1 to n do begin a[i,j]:= 100+random(900);

write(' a[',i,';',j,']:= ' , a[i,j]); end; writeln(' '); end; writeln(' Другий масив:');

for i:=1 to n do begin for j:=1 to n do begin b[i,j]:= 100+random(900);

write(' b[',i,';',j,']:= ', b[i,j]); end; writeln(' '); end; writeln(' ');writeln(' Відповідь.');

fori:=1 to n do beginfor j:=1 to n do begin write(' 3a+3b[', i, ' ,' , j, ']= ', 3*(a[i,j]+b[i,j])); end; writeln(' '); end; writeln(' '); writeln(' '); for k:=1 to n do begin for m :=1 to n do begin write(' 3a-3b[', k, ' ,' , m, ']= ', 3*(a[k,m]- b[k,m])); end; writeln(' '); end; writeln(' '); writeln; end.

Протестувати алгоритм для 10-цифрових: а) const =3; б)const =4; в)const =9.

Практична робота 33.

Алгоритми знаходження максимумів

Завдання 1.Створити та реалізувати алгоритм мовою програмування Pascal, що знаходить найбільше значення виразу *-3x-5y* при таких обмеженнях: *-10<x<10; -10<y<10; 2x-3y<=5; 5x-2y<=8.*

Program MAX1;

var a,b:array[1..1000] of integer; z, k,x,y,i,m,p: integer;

begin k:=0; p:=-1000;

for x:= -9 to 9 do begin writeln(x, ' =x');

```
    for  y:= -9 to 9 do begin   writeln(y, ' =y');
     m:=-3*x-5*y;    writeln(-3*x-5*y,   ' =-3*x-5*y');
       if  ((2*x-3*y)<=5)and((5*x-2*y)<=8)and(m>p)   then    begin
       k:=k+1;    a[k]:=x;     b[k]:=y; writeln(' k= ', k);
 writeln(' a[k]:=', a[k]); writeln(' b[k]:=', b[k]);
  writeln(' значення виразу: - 3*x-5*y =', -3*a[k]-5*b[k]);
      end;  end;   end;
for  i:= 1 to k do  writeln(' x=a[i]= ', a[i], '     y=b[i]= ', b[i], '     i=', i, '
-3*a[i]-5*b[i]=',-3*a[i]-5*b[i], '     5*x-2*y=',5*a[i]-2*b[i],  '     2*x-3*y=', 2*a[i]-3*b[i]);
   z:=-3*a[1]-5*b[1];
for  i:= 1 to k do begin   if      (z<(-3*a[i]-5*b[i]))      then  z:=-3*a[i]-5*b[i]; end;
writeln(' найбільше значення виразу: -3*x-5*y=', z, ' кількість кроків  k=', k);
 writeln(' ***'); end.
```

Завдання 2.Створити та реалізувати алгоритм мовою програмування Pascal, що знаходить найбільше значення виразу *-7x+8y* при таких обмеженнях: *-20<x<20; -5<y<5; x-3y <=12; -x+3y<=10.*

```
Program MAX2;
     var   a,b:array[1..1000] of integer;       z, k,x,y,i,m,p: integer;
begin  k:=0; p:=-1000;
for  x:= -20 to 20 do begin  writeln(x, ' =x');
    for  y:= -5 to 5 do begin   writeln(y, ' =y');
     m:=-7*x+8*y;    writeln(-7*x+8*y,   ' =-7*x+8*y');
       if  ((x-3*y)<=12)and((-x+3*y)<=10)and(m>p)   then    begin
       k:=k+1;    a[k]:=x;     b[k]:=y; writeln(' k= ', k);
 writeln(' a[k]:=', a[k]); writeln(' b[k]:=', b[k]);
  writeln(' значення виразу: - 7*x+8*y =', -7*a[k]+8*b[k]);
      end;  end;   end;
for  i:= 1 to k do  writeln(' x=a[i]= ', a[i], '     y=b[i]= ', b[i], '     i=', i, '
-7*a[i]+8*b[i]=',-3*a[i]-5*b[i], '     x-3*y=',  a[i]-3*b[i],  '     -x+3*y=',  -a[i]+3*b[i]);
```

z:=-7*a[1]+8*b[1];

for i:= 1 to k do begin if (z<(-7*a[i]+8*b[i])) then z:=-7*a[i]+8 *b[i]; end;

writeln(' найбільше значення виразу: -7*x+8*y=', z, ' кількість кроків k=', k);
writeln(' *'); end.**

Практична робота 34.

Алгоритми для квадратних масивів

Задача 1. Випадковим чином задається двомірний масив(це таблиця чисел) , що складається із nxn елементів, якщо n менше 100. Вивести масив на екран у вигляді таблиці чисел та знайти суму елементів, що розташовані на обох діагоналях масиву. Вивести одномірними масивами елементи окремо верхнього і окремо нижнього трикутника матриці без головної діагоналі знайти суму елементів верхнього трикутника та сума елементів нижнього трикутника квадратної матриці nxn.

Розв'язання.

Program Summa_trukutnuk_matriza;

```
const m=100;  n=100;

var  sum,f,s,  d, c:real;    b:array[1..m*n] of real;

 xn:array[1..m*n] of real;     xv:array[1..m*n] of real;      a:array[1..m,1..n] of real;

i,j,p, g, k:integer;

begin

writeln('  Введіть кількість рядків квадратного масиву,  яка менше 100 ');

readln(k);    s:=0;

for i:=1 to k do

for j:=1 to k do  begin

a[i,j]:=int(random*20-10);

b[(i-1)*k+j]:=a[i,j];   end;  writeln('   ');

writeln(' Масив випадкових чисел: ');  writeln('   ');

for i:=1  to k do   begin

for j:=1  to k do  begin      write('  a[',i,';',j,']:= ',a[i,j]  {'  b[',i-1*k+j,']= ',b[(i-1)*k+j]});

end; writeln('   ');end;
```

```
s:=0;   for i:=1 to k do  begin   s:=s+a[i,i];  end;

f:=0;  for i:=1 to k do  begin  f:=f+a[i,k+1-i]; end;

    if   k mod 2 =1 then   sum:=s+f- a[k div 2 +1, k div 2 +1]  else   sum:=s+f;

p:=1;  g:=1;    for i:=1  to k do   begin

for j:=k  downto 1 do  begin

 if   i<j  then  begin

 xv[g]:= a[i,j];  g:=g+1; end;  end; end;

 for i:=1  to k do   begin

 for j:=1  to i-1 do  begin

 if   i>j  then  begin

 xn[p]:= a[i,j];   p:= p+1; end; end;  end;

writeln(' Масив   чисел нижнього трикутника матриці : ');

        d:=0;  for i:=1  to   ((k-1)*k) div 2     do     begin    d:=d+ xn[i];

write('  xn[',i,']= ', xn[i]);end; writeln('   ');

writeln(' Масив   чисел верхнього трикутника матриці:  ');

         c:=0;  for i:=1  to ((k-1)*k) div 2    do   begin

c:=c+ xv[i];  write('  xv[', i, ']= ', xv[i]);   end;    writeln('   ');

writeln('  Сума чисел головної діагоналі матриці:', s); writeln('    ');

writeln('  Сума  чисел  бічної  діагоналі матриці :', f); writeln('    ');

writeln('  Сума чисел на обох діагоналях: ', sum);   writeln('    ');

writeln('  Сума чисел верхнього трикутника матриці ' , c);  writeln('    ');

writeln('  Сума чисел нижнього трикутника матриці  ' ,  d);    writeln('    ');       end.
```

Практична робота 35.

Алгоритми фільтрування елементів масиву

Завдання 1. Петрик П’яточкін вишикував у рядок слоненят та рахує їх по кожному кольору окремо. Всього буває 8 кольорів слоненят. У рядок вишикувались N (N>10)

слоненят. Скільки слоненят кожного кольору стоїть перед Петриком? Створити алгоритм для підрахунку слоненят кожного кольору окремо, пройшовши всього один раз перед строєм. Кольори позначити цифрами від 1 до 8.

```
Program   SLON1;

const k=1000;

var a: array[1..k]  of  integer;       b: array[1..k] of  integer;      i, c, n, m: integer;

begin       n:=1+random(12);       writeln('кількість усіх слонів n=', n,'
');

 writeln('Повна  послідовність  пофарбованих слонів  ');

 for i:=1 to n  do  begin   a[i]:=1+random(8); writeln(i,'ий слон  має колір:   ', a[i],' ');
end; writeln;

 forc:=1 to 8 do    b[c]:=0;  writeln(' Кількість слонів  за  однаковим  кольором  ');

 for i:=1 to n do     b[a[i]]:=b[a[i]]+1;

 for i:=1 to 8 do    writeln(i,'-ий колір має:  ', b[i],'   слонів');  end.
```

Протестувати алгоритм декілька разів.

Завдання 2. Вінні Пух любить складати віршики говорячи речення задом наперед. Якось йому попалось довге складне речення і він забув свій віршик, пробуючи його виговорити. Складіть програму, яка б допомагала ведмедику легко складати такі віршики. Зауваження: віршик може складатись як із 1 слова, так і з декількох, розділених пропусками. В кінці віршика ніколи не ставиться крапка. Довжина віршика менша за 255 символів.

```
Program   VINNI2;

var a,b:string;        i,n:integer;

begin    readln(a);    n:=length(a); b:='  ';

for i:=n downto 1 do    b:=b+a[i];

 writeln ('Це твір-перевертень ВІННІ ПУХА: ',  b);  end.
```

Протестувати алгоритм декілька разів ввівши свій вірш в програму.

Завдання 3. Кіт Леопольд пішов на рибалку та наловив риби. Кожну рибу він старанно зважив. Перша риба (найменша), яку він зважував важила рівно L грам. Кожна наступна рибина була на K грамів важча за попередню. Скільки заважила вся риба, яку наловив Леопольд, якщо відомо, що спіймав він N (N>0) риб?

```
Program   LEOPOLD3;
```

```
var    i, n, L, k, s, t :integer;

begin     n:=2+random(12);       writeln('кількість усіх риб  n=', n,'   ');

L:=2+random(12);       writeln('вага найменшої рибини  L=', L,'  грам ');

k:=3+random(12); writeln(' найменша різниця між вагою двох риб k=', k, '  грам');

 s:=0; t:=l;

 for i:=1 to n do   begin       s:=s+t;      t:=t+k;    end;     writeln (s);   end.
```

Протестувати алгоритм декілька разів.

Практична робота 36.

Алгоритми сортування чисел способом «Бульбашкою»

Завдання 1.Реалізувати алгоритм сортування методом «Бульбашкою», що використовує чотири допоміжні процедури та функції.

```
program sorting;                                  { назва  алгоритму }

const maxcnt=5;                                  { назва  постійної довжини масиву  чисел}

type vector=array[1..maxcnt] of real;          {назва динамічного масиву із дійсних чисел  }

var cmpcnt,swpcnt:  integer;                     {назва змінних величин:  цілих чисел }

    trace:  boolean;                             {назва змінних величин:  логічних (так/ні) }

procedure writevector(n: integer; m: vector);  {допоміжна процедура друкування масиву}

var i: integer;                                   {назва змінних величин:  цілих чисел }

begin     writeln('   це m[1]  елемент відсортованого рядка' , m[1]:6:2);   {цикл друку}

 for i:=2 to n do writeln('  це m[', i, '] -ий елемент відсортованого рядка ', m[i]:6:2);

 writeln;  end;                                  {закінчення процедури друкування елементів масиву }

function   less(a, b: real): boolean;            {допоміжна функціяпорівняння елементів}

begin      cmpcnt:=cmpcnt+1;                    { неповне розгалуження для  порівняння чисел }

 If  trace   then  writeln('Порівняння', a:6:2, ' з ', b:6:2);   less:=a<b;  end;

 procedure swap(var a,b:real);                    {допоміжна процедура - обмін елементів }

var c:real;                                      {назва змінних величин:  дійсних  чисел }
```

begin swpcnt:=swpcnt+1; c:=a; a:=b; b:=c end; {виконується обмін елементів }

procedure bubble(n: integer; var m: vector); {**процедура** сортування - "Бульбашка" }

var i,j:integer; {назва змінних величин: цілих чисел }

begin

for i:=n-1 downto 1 do {цикл з лічильником від більшого до меншого }

for j:=1 to i do {вкладений цикл з лічильником від меншого до більшого }

if less(m[j+1],m[j]) then begin { неповне розгалуження з використанням логік-функції}

swap(m[j],m[j+1]); {використання процедури обміну елементами }

if trace then writevector(n,m) end; end; { процедура друку елементів }

{Основний алгоритм, який використовує допоміжні алгоритми}

var n,i: integer; {назва змінних величин: цілих чисел }

m: vector; {назва змінних величин: масиву чисел }

ans: char; {назва змінних величин: буквених символів }

begin

repeat {цикл з післяумовою для введення довжини масиву }

write('Введено розмір масиву(від 1 до ', maxcnt, '): ');

n:=1+random(5); writeln('n=', n); {випадкове ціле чисел – це довжина масиву}

until (1<=n) and (n<=maxcnt); {Умова перевірки закінчення циклу}

writeln('Введено масив випадкових чисел:');

for i:=1 to n do begin {цикл з лічильником для введення елементів масиву }

m[i]:=2+random(159); writeln('Введено випадкове число:', m[i]);

writeln; end; {цикл з лічильником для виведення елементів масиву }

write('Показати процес роботи? (y/N) '); readln(ans); {діалог з користувачем}

trace:=(ans='y')or(ans='Y');

writeln('Обрано метод сортування "Бульбашка":'); cmpcnt:=0; swpcnt:=0;

bubble(n,m); {виклик процедури сортування «бульбашкою» чисел }

writeln('Всього виконано ', cmpcnt, ' порівнянь та ', swpcnt, ' перестановок');

writeln(' Відсортований масив в порядку зростання:'); writevector(n,m); end.

Практична робота 37.

Алгоритми сортування масиву різними способами з процедурами

Завдання 1. Реалізувати алгоритм сортування методом Хоора, що використовує чотири допоміжні процедури та функції.

program sorting; { назва алгоритму }

const maxcnt=5; { назва постійної довжини масиву чисел}

type vector=array[1..maxcnt] of real; {назва динамічного масиву із дійсних чисел }

var cmpcnt,swpcnt: integer; {назва змінних величин: цілих чисел }

trace: boolean; {назва змінних величин: логічних (так/ні) }

procedure writevector(n: integer; m: vector); {допоміжна **процедура** друкування масиву}

var i: integer; {назва змінних величин: цілих чисел }

begin writeln(' це m[1] елемент відсортованого рядка' , m[1]:6:2); {цикл друку}

for i:=2 to n do writeln(' це m[', i, '] -ий елемент відсортованого рядка ', m[i]:6:2);

writeln; end; {закінчення процедури друкування елементів масиву }

function less(a, b: real): boolean; {допоміжна **функція**порівняння елементів}

begin cmpcnt:=cmpcnt+1; { неповне розгалуження для порівняння чисел }

If trace then writeln('Порівняння', a:6:2, ' з ', b:6:2); less:=a<b; end;

procedure swap(var a,b:real); {допоміжна **процедура** - обмін елементів }

var c:real; {назва змінних величин: дійсних чисел }

begin swpcnt:=swpcnt+1; c:=a; a:=b; b:=c end; {виконується обмін елементів }

procedure hoor(a,b: integer; var m: vector); { Рекурсивний алгоритм швидкого сортування Хоора}

var i,j,c: integer; k: real;

begin if a<b then begin c:=(a+b) div 2; k:=m[c];{Обираємо ключовий елемент}

{ ділимо масив на дві групи, порівнюючи з ключовим елементом }

i:=a; j:=b; repeat if (i=c) and (i<j) then begin swap(m[i],m[j]); c:=j end;

if not less(k,m[i]) then i:=i+1 else begin swap(m[i],m[j]); j:=j-1;

if j=c then c:=i end until i>j;

{ Якщо друга група порожня – перносимо в неї ключовий елемент }

if i>b then i:=b; if trace then writevector(b,m);

{виконуємо сортування кожної частини, викликаючи саму себе процедури}

hoor(a,i-1,m); hoor(i,b,m); end; end;

{Основний алгоритм, який використовує допоміжні алгоритми}

var n,i: integer; m: vector; ans: char; {назва змінних величин}

begin repeat {цикл з післяумовою для введення довжини масиву }

write('Введено розмір масиву(від 1 до ', maxcnt, '): '); n:=2+random(5); writeln('n=', n);

until (1<=n) and (n<=maxcnt); {Умова перевірки закінчення циклу}

writeln('Введено масив випадкових чисел:');

for i:=1 to n do begin {цикл з лічильником для введення елементів масиву }

m[i]:=2+random(159); writeln('Введено випадкове число:', m[i]);

writeln; end; {цикл з лічильником для виведення елементів масиву }

write('Показати процес роботи? (y/N) '); readln(ans); {діалог з користувачем}

trace:=(ans='y')or(ans='Y');

writeln('Обрано метод сортування Хоора'); cmpcnt:=0; swpcnt:=0;

hoor(1,n,m); {виклик процедури сортування методом Хоора чисел}

writeln('Всього виконано ', cmpcnt, ' порівнянь та ', swpcnt, ' перестановок');

writeln(' Відсортований масив в порядку зростання:'); writevector(n,m); end.

Завдання 2. Реалізувати алгоритм сортування різними методами.

program sorting;

```
const maxcnt=5;
type vector=array[1..maxcnt] of real;
var cmpcnt,swpcnt: integer;
  trace: boolean;

{  Допоміжна процедура друкування масиву }

procedure writevector(n: integer; m: vector);
var i: integer;
begin
  writeln('це перший елемент відсортованого рядка' , m[1]:6:2);
  for i:=2 to n do
writeln('  це m[', i, '] -ий елемент відсортованого рядка ', m[i]:6:2);
  writeln;
end;

  { Допоміжна функція порівняння елементів }
Function   less(a, b: real): boolean;
begin
  cmpcnt:=cmpcnt+1;
  if trace then writeln('Порівняння', a:6:2, ' з ', b:6:2);
  less:=a<b;
end;

  { Допоміжна процедура – обмін елементів }
procedure swap(var a,b:real);
var c:real;
```

```
begin
  swpcnt:=swpcnt+1;
  c:=a; a:=b; b:=c
end;


   { Найпростіший алгоритм сортування  - "бульбашка" }
procedure bubble(n: integer; var m: vector);
var i,j:integer;
begin
  for i:=n-1 downto 1 do
    for j:=1 to i do
      if less(m[j+1],m[j]) then begin
            swap(m[j],m[j+1]);
            if trace then writevector(n,m)
      end;
end;
  { Метод мінімального елемента мінімізує кількість перестановок }
procedure minelement(n: integer; var m: vector);
var i,j,k:integer;
begin
  for i:=1 to n-1 do begin
  { Знаходимо мінімальний елемент  з тих, що залишилось … }
    k:=i;
    for j:=i+1 to n do
      if less(m[j],m[k]) then k:=j;
  { і ставимо його на місце }
    if i<>k then begin
```

```
      swap(m[i],m[k]);
      if trace then writevector(n,m)
    end;
  end;
end;
```

{Метод вставки зменшує кількість порівнянь до $n*\log_2(n)$ }

```
procedure inserting(n: integer; var m: vector);
var i,a,b,c:integer;
  k: real;
begin
  for i:=2 to n do begin
    { У відсортованій частині  масиву шукаємо місце для чергового елемента}
    { метод ділення навпіл }
    a:=1; b:=i;
    k:=m[i];
    while a<b do begin
      c:=(a+b)div 2;
      if less(k,m[c]) then
            while b>c do begin
              m[b]:=m[b-1];
              b:=b-1
            end
      else a:=c+1
    end;
    m[a]:=k; swpcnt:=swpcnt+i-a;
    if trace then writevector(n,m);    end;  end;
```

```
{Пірамідальне сортування – один з найефективніших алгоритмів}
procedure pyramidal(n: integer; var m: vector);
label 1,2;
var i,j,k,l: integer;
  v: real;
begin
  { Перший етап: побудова піраміди }
  if trace then writeln('Перший етап');
  for i:=2 to n do begin
    j:=i;
    repeat
      k:=j div 2;
      if less(m[k],m[j]) then swap(m[k],m[j])
      else goto 1;
      j:=k
    until j=1;
  1: if trace then writevector(n,m)
  end;
  { Тепер виконується умова m[i/2]>=m[i] для будь-якого  i>1 }
  { Другий етап: вилучаємо з піраміди вершину і переносимо в кінець масиву}
  if trace then writeln('Другийетап');
  for i:=n downto 2 do begin
    v:=m[1];
    j:=1;
    while 2*j<i do begin
      k:=2*j;
```

```
      if less(m[k],m[k+1]) then begin
            m[j]:=m[k+1];
            j:=k+1
      end else begin
            m[j]:=m[k];
            j:=k
      end;
      swpcnt:=swpcnt+1
    end;
    m[j]:=m[i];
    if 2*j>i then
      while j>1 do begin
            l:=j div 2;
            if less(m[j],m[l]) then goto 2;
            swap(m[l],m[j]);
            j:=l
    end;
    2: m[i]:=v; swpcnt:=swpcnt+1;
    if trace then writevector(n,m)
  end;
end;
```

{ Рекурсивний алгоритм швидкого сортування Хоора}

```
procedure hoor(a,b: integer; var m: vector);
var   i,j,c: integer;     k: real;
begin     if a<b then   begin      { Обираємо ключовий елемент }
   c:=(a+b) div 2;       k:=m[c];
   { ділимо масив на дві групи, порівнюючи з ключовим елементом }
```

```
  i:=a; j:=b;
  repeat
    if (i=c) and (i<j) then begin swap(m[i],m[j]); c:=j end;
    if   not less(k,m[i])   then   i:=i+1
    else  begin
          swap(m[i],m[j]);        j:=j-1;
          if j=c then c:=i
    end
  until i>j;
 {Якщо друга група порожня – перносимо в неї ключовий елемент}
  if   i>b   then   i:=b;
  if   trace   then   writevector(b,m);
  {виконуємо сортування кожної частини}
  hoor(a,i-1,m);        hoor(i,b,m);     end;   end;


var
  n,i: integer;
  m: vector;
  ans: char;
begin
  repeat
    write('Введено розмір масиву(від 1 до ', maxcnt, '): ');
    maxcnt:=5+random(5)
  until (1<=n) and (n<=maxcnt);
  writeln('Введено масив випадкових чисел:');
  for i:=1 to n do
    m[i]:=2+random(159);
```

```
  readln;
  write('Показати процес роботи? (y/N) '); readln(ans);
  trace:=(ans='y')or(ans='Y');
  writeln('оберіть метод сортування:');
  writeln('1 - "Бульбашка"');
  writeln('2 – мінімальний елемент');
  writeln('3 – метод вставки');
  writeln('4 – пірамідальне сортування');
  writeln('5 – швидке сортування Хоора');
  cmpcnt:=0; swpcnt:=0;
  repeat
    readln(ans);
    case ans of
      '1': bubble(n,m);
      '2': minelement(n,m);
      '3': inserting(n,m);
      '4': pyramidal(n,m);
      '5': hoor(1,n,m);
    end;
  until (ans>='1') and (ans<='5');
  writeln('Всього виконано ', cmpcnt, ' порівнянь та ', swpcnt, ' перестановок');
  writeln(' Відсортований масив:');
  writevector(n,m)
end.
```

1. БАНК ЗАВДАНЬ

1.1 Банк завдань на програмування лінійних алгоритмів

1. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника a, b, c (це три дійсні числа – real, які вводяться з клавіатури) половину периметра(p=a+b+c), площу($S=(p(p-a)(p-b)(p-c))^{0,5}$), радіус описаного кола (R=abc/4S), радіус вписаного кола(r=S/p). Вивести на екран шукані величини.

2. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника a, b, c (це три дійсні числа – real, які вводяться з клавіатури) усі висоти($H_a = 2(p(p-a)(p-b)(p-c))^{0,5}/a$, $H_b = 2(p(p-a)(p-b)(p-c))^{0,5}/b$, $H_c = 2(p(p-a)(p-b)(p-c))^{0,5}/c$) трикутника. Вивести на екран шукані величини.

3. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника a, b, c (це три дійсні числа – real, які вводяться з клавіатури) усі медіани ($m_a = 0.5(2b^2+2c^2-a^2)^{0,5}$, $m_b = 0.5(2a^2+2a^2-b^2)^{0,5}$), $m_c = 0.5(2b^2+2a^2-c^2)^{0,5}$ трикутника. Вивести на екран шукані величини.

4. Створити та реалізувати мовою програмування лінійний алгоритм, що за відомими фізичними величинами a_1, m_1, m_2 *(дійсні числа)* і законом збереження імпульсу $a_2=a_1 m_1/m_2$ знаходить прискорення другого фізичного об'єкта(тіла). Вивести на екран шукані величини.

5. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за відомим ребром куба *a(дійсне число)* знаходить площу поверхні куба($S = 6a^2$), об'єм куба ($V = a^3$), діагональ куба ($D=a(3)^{0.5}$), діагональ грані куба ($L=a(2)^{0.5}$). Вивести на екран шукані величини.

6. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

7. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

8. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

9. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

10. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

11. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

12. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

13. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

14. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

15. Створити та реалізувати мовою програмування лінійний алгоритм, що знаходить за трьома відомими сторонами трикутника периметр, площу, радіус вписаного кола, радіус описаного кола. Вивести на екран шукані величини.

1.2 Банк завдань на програмування алгоритмів на одномірних масивах

1 частина.

Складіть і реалізуйте алгоритми для розв’язування поданих нижче задач.

1. Задано одновимірний масив з 5 випадкових цілих чисел. Замінити всі елементи даного масиву на протилежні за знаком. Вивести на екран елементи даного та зміненого масивів.

2. Задано одновимірний масив з 7 випадкових дійсних чисел. Замінити всі елементи даного масиву на їхні квадрати. Вивести на екран елементи даного та зміненого масивів.

3. Задано одновимірний масив з 8 випадкових цілих чисел. Збільшити вдвічі всі елементи даного масиву. Вивести на екран елементи даного та зміненого масивів.

4. Задано одновимірний масив з 8 випадкових цілих чисел. Підрахувати кількість від’ємних елементів у даному масиві. Вивести на екран елементи даного масивів та кількість чисел, що відповідають умові..

5. Задано одновимірний масив з 18 випадкових дійсних чисел різних знаків. Підрахувати кількість додатних елементів у даному масиві. Вивести на екран елементи даного масивів та кількість чисел, що відповідають умові..

6. Задано одновимірний масив з 10 випадкових дійсних чисел різних знаків. Підрахувати суму від’ємних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

7. Задано одновимірний масив з 11 випадкових дійсних чисел різних знаків. Підрахувати

суму додатних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

8. Задано одновимірний масив з 14 випадкових дійсних чисел різних знаків. Підрахувати середнє арифметичне всіх елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

9. Задано одновимірний масив з 18 випадкових дійсних чисел різних знаків. Підрахувати середнє арифметичне додатних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

10. Задано одновимірний масив з 16 випадкових дійсних чисел різних знаків. Підрахувати середнє арифметичне від'ємних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

11. Задано одновимірний масив з 12 випадкових дійсних чисел різних знаків. Поміняйте місцями такі елементи масиву: 1-й на 9-й, 2-й на 8-й і т. ін. Вивести на екран елементи даного масивів та чотири числа, що обмінюються згідно умові..

12. Задано одновимірний масив з 16 випадкових дійсних чисел різних знаків. Знайти найбільший елемент масиву. Вивести на екран елементи даного масивів та число, що відповідає умові..

13. Задано одновимірний масив з 12 випадкових дійсних чисел різних знаків. Знайти найменший елемент масиву. Вивести на екран елементи даного масивів та число, що відповідає умові..

14. Задано одновимірний масив з 13 випадкових дійсних чисел різних знаків. Знайти різницю між найбільшим і найменшим елементом. Вивести на екран елементи даного масивів та число, що відповідає умові..

15. Задано одновимірний масив з 10 випадкових дійсних чисел різних знаків. Знайти суму кубів від'ємних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

16. Задано одновимірний масив з 14 випадкових дійсних чисел різних знаків. Знайти суму кубів додатних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

17. Задано одновимірний масив з 10 випадкових дійсних чисел різних знаків. Знайти суму квадратів від'ємних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

18. Задано одновимірний масив з 15випадкових дійсних чисел різних знаків. Знайти суму квадратів додатних елементів. Вивести на екран елементи даного масивів та число, що відповідає умові..

19. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. З'ясувати, чи є у масиві елемент, що дорівнює 7. Якщо є, то визначити, який його порядковий номер (індекс). Вивести на екран елементи даного масивів та число, що відповідає умові..

20. Задано одновимірний масив з 14 випадкових дійсних чисел різних знаків. У даному

масиві всі елементи, що більші від числа 1, замінити на 0. Вивести на екран елементи даного масивів та число, що відповідає умові..

21. Задано одновимірний масив з 13 випадкових дійсних чисел різних знаків. У даному масиві всі елементи, що більші від числа 5, замінити на 1. Вивести на екран елементи даного масивів та число, що відповідає умові..

22. Задано одновимірний масив з 12 випадкових дійсних чисел різних знаків. Знайти суму елементів, що мають парні індекси. Вивести на екран елементи даного масивів та число, що відповідає умові..

23. Задано одновимірний масив з 11 випадкових дійсних чисел різних знаків. Знайти суму елементів, що мають непарні індекси. Вивести на екран елементи даного масивів та число, що відповідає умові..

24. Задано одновимірний масив з 17 випадкових дійсних чисел різних знаків. Всі елементи даного масиву замінити на їхні куби. Вивести на екран елементи даного масивів та числа, що відповідають умові..

25. Задано одновимірний масив з 17 випадкових дійсних чисел різних знаків. Елементи, що більші від числа 3, замінити на їхні куби. Вивести на екран елементи даного масивів та числа, що відповідають умові.

26. Задано одновимірний масив з 16 випадкових дійсних чисел різних знаків. Підрахувати кількість елементів, що більші від числа 3. Вивести на екран елементи даного масивів та числа, що відповідають умові.

27. Задано одновимірний масив з 15 випадкових дійсних чисел різних знаків. Підрахувати кількість елементів, що не більші від числа -1. Вивести на екран елементи даного масивів та числа, що відповідають умові.

28. Задано одновимірний масив з 14 випадкових дійсних чисел різних знаків. Підрахувати середнє арифметичне елементів, що не менші від 1. Вивести на екран елементи даного масивів та числа, що відповідають умові.

29. Задано одновимірний масив з 13 випадкових дійсних чисел різних знаків. Підрахувати число елементів, що лежать в інтервалі (-3; 8). Вивести на екран елементи даного масивів та числа, що відповідають умові.

30. Задано одновимірний масив з 12 випадкових дійсних чисел різних знаків. Скласти з масиву Кмасив А, що містить тільки позитивні елементи масиву К. Вивести на екран елементи даного масивів та числа, що відповідають умові.

31. Задано одновимірний масив з 11 випадкових дійсних чисел різних знаків. Скласти з масиву Кмасив А, що містить усі елементи масиву К, які за величиною перевищують їхнє середнє арифметичне. Вивести на екран елементи даного масивів та числа, що відповідають умові.

32. Задано одновимірний масив з 10 випадкових дійсних чисел різних знаків. Побудувати масив А, що містить усі масиви елементу К, які задовольняють таку умову: $-4 < A_i < 10$. Вивести на екран елементи даного масивів та числа, що відповідають умові..

33. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. Побудувати масив А, що містить усі парні елементи масиву К. Вивести на екран елементи даного масивів та числа, що відповідають умові.

34. Обчислити середнє арифметичне всіх чисел масиву К, що мають непарні номери. Вивести на екран елементи даного масивів та числа, що відповідають умові.

35. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. Замінити елементи даного масиву на їхні модулі. Вивести на екран елементи даного масивів та числа, що відповідають умові.

36. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. Замінити додатні елементи на число 5, а від'ємні - на число 7. Вивести на екран елементи даного масивів та числа, що відповідають умові.

37. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. Заповніть новий масив, збільшивши усі числа даного масиву на 2. Вивести на екран елементи даного масивів та числа, що відповідають умові.

38. Задано одновимірний масив з 10 випадкових цілих чисел різних знаків. Додати до від' ємних елементів даного масиву по 1, а додатні елементи зменшити на 1. Вивести на екран елементи даного масивів та числа, що відповідають умові.

2 частина.

Складіть програми для розв'язання поданих нижче задач,записуючи розв'язки в одновимірні масиви.

1. 1000 шт. цегли можна перевозити візками місткістю 100, 300, 400, 500 шт. цегли. Отримати всі можливі варіанти перевезень. Підрахувати їхню кількість.

2. Футбольний м'яч коштує 65 грн. Отримати всі можливі варіанти оплати,якщо у покупця є 5-,10-,20-гривневі купюри. Підрахувати кількість варіантів.

3. Садівникові потрібно 18 кг. Мінеральних добрив. Отримати всі можливі варіанти купівлі добрива,якщо в магазині продаються розфасовки по 5-,4-,2-кг. Підрахувати кількість варіантів.

4. Повітроплавцеві потрібно заповнити воднем повітряну кулю місткістю 17 куб. м балончиками по 1-,2-,5-куб. м водню. Отримати всі можливі варіанти наповнення. Підрахувати їхню кількість.

5. Шляховим майстрам потрібно прокласти 190 м. залізниці рейками по 8 і 10 м. Отримати всі можливі варіанти прокладання. Підрахувати їхню кількість.

6. 14 літрів соку потрібно розлити в 4-,3-,2- та 1-літрові банки. Отримати всі можливі варіанти розливу. Підрахувати їхню кількість.

7. 36 кг яблук потрібно розфасувати у пакети по 2-,4-,5- і 10-кг. Отримати усі можливі варіанти розфасування. Підрахувати їхню кількість.

8.20 кг яблук потрібно розфасувати у пакети по 2-,4- кг. Отримати всі можливі варіанти розфасування. Підрахувати їхню кількість.

9. 240 екскурсантів можна розсадити в автобуси ЛАЗ(місткість-40 осіб) і ПАЗ(місткість-30 осіб). Отримати всі можливі варіанти замовлень автобусів для перевезення екскурсантів. Підрахувати кількість варіантів.

10. 13 літрів соку потрібно розлити в 4-,2- та 1-літрові банки. Вивести на монітор всі можливі варіанти розливу. Підрахувати їхню кількість.

11. Друкарці можна друкувати книги обсягом 30, 40 і 60 аркушів за її вибором. Усього вона надрукувала 1200 аркушів. Визначити, скільки і яких книг вона могла надрукувати. Отримати всі можливі варіанти. Підрахувати їхню кількість.

12. Для ремонту дороги потрібно завезти 24 т щебеню. В автопарку є самоскиди вантажопідйомністю 3, 4 і 6 т. Отримати всі можливі варіанти перевезення щебеню. Підрахувати їхню кількість.

13. 600 шт. цегли можна перевозити возами місткістю 60 і 45 цеглин. Отримати усі можливі плани перевезення. Підрахувати їхню кількість.

14. 200 літрів бензину потрібно розлити в баки місткістю 60, 45, і 25 л. Отримати всі можливі варіанти розливу. Підрахувати їхню кількість.

1.3 Банк завдань на програмування алгоритмів

на двовимірних масивах

1. Задано двовимірний масив 2х2 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що замінює всі елементи даного масиву на взаємно обернені числа, окрім нульових елементів. Вивести на екран елементи даного та зміненого масивів.

2. Задано двовимірний масив 3х3 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що замінює елементи з парними індексами з даного масиву на потроєні числа, окрім нульових елементів. Вивести на екран елементи даного та зміненого масивів.

3. Задано двовимірний масив 4х4 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що замінює елементи з непарними індексами даного масиву на квадрати суми індексів цього елемента, окрім одиничних елементів масиву. Вивести на екран елементи даного та зміненого масивів.

4. Задано двовимірний масив 2х3 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, який підраховує кількість та суму невід’ємних елементів даного масиву. Вивести на екран елементи даного масиву та лінійного масиву

з невід’ємними елементами.

5. Задано двовимірний масив 4х2 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, який підраховує кількість та суму недодатних елементів даного масиву. Вивести на екран елементи даного масиву та лінійного масиву з недодатними елементами.

6. Задано двовимірний масив 5х5 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що підраховує кількість елементів даного масиву, які належать проміжку (1; 5), окрім цілих чисел. Вивести на екран елементи даного та лінійного масиву з недодатними елементами.

7. Задано двовимірний масив 5х5 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що підраховує середнє арифметичне елементів головної діагоналі даного масиву, які належать проміжку (-4; 0), окрім цілих чисел. Вивести на екран елементи даного та лінійного масиву, що утворений з елементів головної діагоналі, які належать проміжку (-4; 0),

8. Задано двовимірний масив 2х3 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що підраховує кількість елементів даного масиву, які не належать проміжку (0,5; 3,1), окрім цілих чисел. Вивести на екран елементи даного та лінійний масив, що утворений з шуканих елементів, які не належать проміжку (0,5; 3,1).

9. Задано двовимірний масив 2х3 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що підраховує кількість елементів даного масиву, які належать проміжку (1; +оо), окрім цілих чисел. Вивести на екран елементи даного та лінійного масиву, що утворений із шуканих елементів, які належать проміжку (1; +оо).

10. Задано двовимірний масив 5х5 з випадкових дійсних чисел. Створити та реалізувати мовою програмування алгоритм, що підраховує суму квадратів від’ємних елементів даного масиву, які належать проміжку (-оо; 5), окрім цілих чисел. Вивести на екран елементи даного та зміненого масивів.