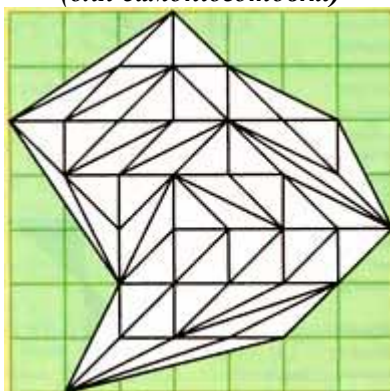


*Методи складання алгоритмів та їх аналіз*  
**Обчислювальна геометрія**

*(для самопідготовки)*



Луцьк –2005

Посібник містить теоретичний матеріал та приклади розв'язаних задач для самостійного вивчення наступної теми з курсу програмування: “Обчислювальна геометрія та числові методи”

Програми розв'язку задач реалізовано в мові програмування Паскаль.

*Для учнів класів з поглибленим вивченням програмування*

Відгуки та пропозиції надсилати за адресою:

**263000 м. Луцьк, вул. Черняхівського, 8, гімназія №14, кабінет № 23**

**Укладач:** *вчитель інформатики*  
*НВК «Гімназія №14» І.В.Гісь*

**Рецензент:** *методист кабінету інформатики*  
*ВОШОПП Демкова Л.М.*

# Зміст

ст.

<b>Вступ</b> .....	
<b>1. Визначення площі многокутника</b> .....	
Задача 1. Визначення довжини відрізка .....	
Задача 2. Визначення площі трикутника (формула Герона) .....	
Задача 3. Визначення площі трикутника за заданими координатами вершин .....	
Задача 4. Визначення площі опуклого многокутника .....	
Задача 5. Визначення площі довільного многокутника .....	
<b>2. Визначення положення точки відносно n-кутника:</b>	
Задача 1. Перевірка належності точки прямій .....	
Задача 2. Перевірка належності точки відрізку .....	
Задача 3. Перевірка належності точки трикутнику .....	
Задача 4. Перевірка належності точки многокутнику .....	
<b>3. Визначення опуклої оболонки</b> .....	
<b>4. Задачі на обчислювальну геометрію</b> .....	
<b>Література</b> .....	

## **Вступ**

На початку 70-х років Н.Вірт розробив мову програмування Паскаль, призначену для початкового навчання структурному програмуванню. Вона була створена на базі мови Алго-60. Поява надійних трансляторів з мови Паскаль, що займають невеликий обсяг пам'яті, привела до поширення її серед різних категорій користувачів.

У багатьох навчальних закладах навчання програмуванню починається з мови Паскаль. Це зумовлюється такими особливостями мови:

1. Простота і природність мови.
2. Побудова мови за принципами структурного програмування та структурованої організації даних.
3. Здійснення розробки трансляторів з мови Паскаль за допомогою доступних і добре опрацьованих методик.
4. Транслятор дозволяє оптимізувати програми.
5. Конструкції навчальної алгоритмічної мови аналогічні до відповідних операторів Паскаль.

Популярність мови особливо зросла після створення трансляторів для персональних комп'ютерів. Виникли численні версії мови. Найпоширенішою є діалект мови, розроблений фірмою Borland International, Inc під назвою Turbo Pascal. Ця версія сумісна з міжнародним стандартом ISO Pascal Standard 1982. Вона була поширена на 8 і 16 – розрядних персональних комп'ютерах. Для 32-розрядних нових комп'ютерів використовуються новіші версії Borland Pascal, Free Pascal, Delphi (в основі лежить Object Pascal).

Посібник містить опис алгоритмів та програм, що реалізують математичні методи обчислювальної геометрії. Так як згідно Програми для загальноосвітніх навчальних закладів (Навчальні програми для профільного навчання, Програми факультативів, спецкурсів, пропедевтичних

курсів, гуртків, Програми для загальноосвітніх навчальних закладів, спеціалізованих шкіл, гімназій, ліцеїв,

Інформатика та програмування 8-11 класи (Автори: Голубніча Н.В., Караванова Т.П., Костюков В.П.) пропонується викладати теми: **Методи складання алгоритмів та їх аналіз**, яка містить наступні розділи:

1. Методика побудови алгоритмів, оцінка їх ефективності.
2. Структури даних.
3. Пошукові алгоритми.
4. Методи сортування.
5. Обчислювальна геометрія та числові методи.
6. Застосування комбінаторики для розв'язання задач.
7. Основи теорії графів.
8. Основи лінійного програмування.
9. Основи динамічного програмування.
10. “Жадібні” алгоритми.

При вивченні теми «Обчислювальна геометрія та числові методи» учні повинні знати:

- сутність векторного добутку та напрямку повороту;
- сутність умов перетину відрізків;
- алгоритми визначення площі простого многокутника, положення точки відносно простого многокутника, опуклої оболонки, пари найближчих точок, пари найвіддаленіших точок;
- алгоритм метода виключення для розв'язання алгоритмічних задач.

Учні повинні мати уявлення про:

- застосування поняття векторного добутку для реалізації алгоритмів розв'язання геометричних задач.

Учні повинні вміти:

- застосовувати алгоритми визначення площі простого многокутника, положення точки відносно простого многокутника, опуклої оболонки, пари найближчих точок, пари найвіддаленіших точок для реалізації конкретних задач;

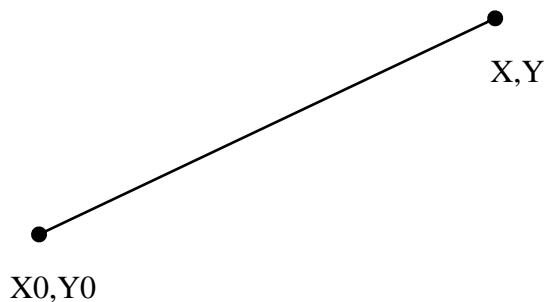
застосовувати алгоритм метода виключення для реалізації конкретних задач.

Як раз свій варіант викладання теми «Обчислювальна геометрія та числові методи» я пропоную в даному посібнику з використанням опису підпрограм.

## 1. Визначення площі многокутника

### Задача 1. Визначення довжини відрізка

За заданими координатами початку і кінця відрізка визначити його довжину.



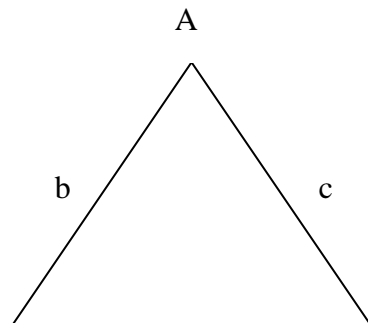
Довжина відрізка  $L = \sqrt{(x - x_0)^2 + (y - y_0)^2}$

Процедура визначення довжини:

```
procedure line(x0,y0,x,y:real;var l:real);  
begin  
l:=sqrt(sqr(x-x0)+sqr(y-y0));  
end;
```

### Задача 2. Визначення площі трикутника (формула Герона)

За заданими сторонами трикутника визначити площу трикутника.



Площа трикутника за формулою Герона обчислюється:

$$S = \sqrt{p(p-a)(p-b)(p-c)}, \text{ де } p = \frac{a+b+c}{2}$$

Процедура визначення площі трикутника:

```

procedure pl(a,b,c:real; var s:real);
    var p:real;
begin
    p:=(a+b+c)/2;
    s:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

```

### Задача 3. Визначення площі трикутника за заданими координатами вершин

За заданими координатами вершин трикутника визначити його площу.

Для визначення площі скористуємось двома процедурами за координатами знаходимо довжини сторін за довжинами сторін визначаємо площу трикутника.

Програмний код:

```

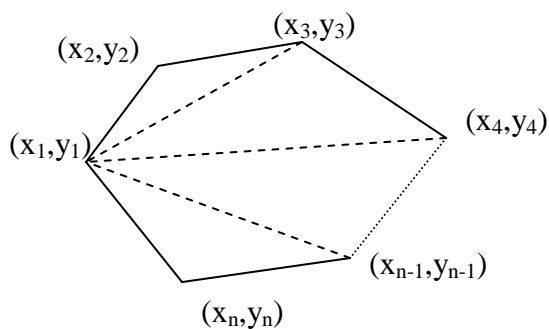
Program pr1;
    var x1,y1,x2,y2,x3,y3,st:real;
procedure line(x0,y0,x,y:real;var l:real);
begin
    l:=sqrt(sqr(x-x0)+sqr(y-y0));
end;
procedure pl(a,b,c:real; var s:real);
var p:real;
begin
    p:=(a+b+c)/2;
    s:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
readln(x1,y1,x2,y2,x3,y3);
line(x1,y1,x2,y2,a);
line(x2,y2,x3,y3,b);
line(x3,y3,x1,y1,c);
pl(a,b,c,st);
writeln (st);
end.

```

### Задача 4. Визначення площі опуклого багатокутника

За заданими координатами вершин багатокутника визначити його площу.



Розбиваємо багатокутник на трикутники і знаходимо суму трикутників. Координати вершин багатокутника заносимо в масив.

Програмний код:

**Program Prog 2;**

```

var xx,yy:array[1..100] of real;
a1,b1,c1,st,s_mn:real;
i,n:integer;
procedure line(x0,y0,x,y:real;var l:real);
begin
l:=sqrt(sqrt(x-x0)+sqrt(y-y0));
end;

```

**procedure pl(a,b,c:real; var s:real);**

```

var p:real;
begin
p:=(a+b+c)/2;
s:=sqrt(p*(p-a)*(p-b)*(p-c));
end;
begin
readln(n);
for i:=1 to n do readln(xx[i],yy[i]);
s_mn:=0;
for i:=1 to n-2 do begin
line(xx[1],yy[1],xx[i+1],yy[i+1],a1);
line(xx[i+1],yy[i+1],xx[i+2],yy[i+2],b1);
line(xx[i+2],yy[i+2],xx[1],yy[1],c1);
pl(a1,b1,c1,st);
s_mn:=s_mn+st;
end;
writeln(s_mn);
end.
end.

```

**Задача 5. Визначення площі довільного багатокутника**

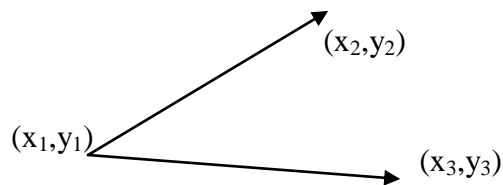
За заданими координатами вершин багатокутника визначити його площу.  
Для обчислення площі можна використати формулу:

$$S = \frac{1}{2} | (x_1 y_2 - x_2 y_1) + (x_2 y_3 - x_3 y_2) + \dots + (x_n y_1 - x_1 y_n) |$$

Обґрунтування:

1) Для трикутника:

$$S = \frac{1}{2} | (x_1 y_2 - x_2 y_1) + (x_2 y_3 - x_3 y_2) |$$

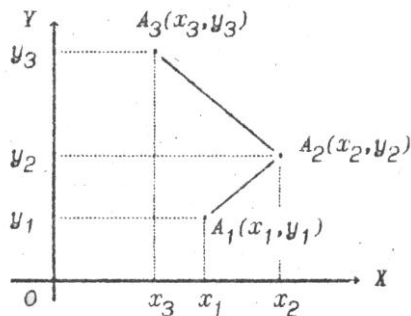


Координати векторів  $(x_2 - x_1, y_2 - y_1)$ ,  $(x_3 - x_1, y_3 - y_1)$ .

Модуль векторного добутку рівний площі паралелограма, а  $\frac{1}{2}$  модуля векторного добутку - площі трикутника.

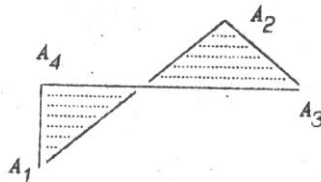
$$\begin{aligned}
S &= \frac{1}{2} \left| \begin{vmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{vmatrix} \right| = \frac{1}{2} | (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1) | = \\
&= \frac{1}{2} | x_2 y_3 - x_2 y_1 - x_1 y_3 + x_1 y_1 - x_3 y_2 + x_3 y_1 + x_1 y_2 - x_1 y_1 | = \\
&= \frac{1}{2} | (x_1 y_2 - x_2 y_1) + (x_2 y_3 - x_3 y_2) + (x_3 y_1 - x_1 y_3) |
\end{aligned}$$

2) Структуру формули можна зрозуміти на прикладі трикутника.



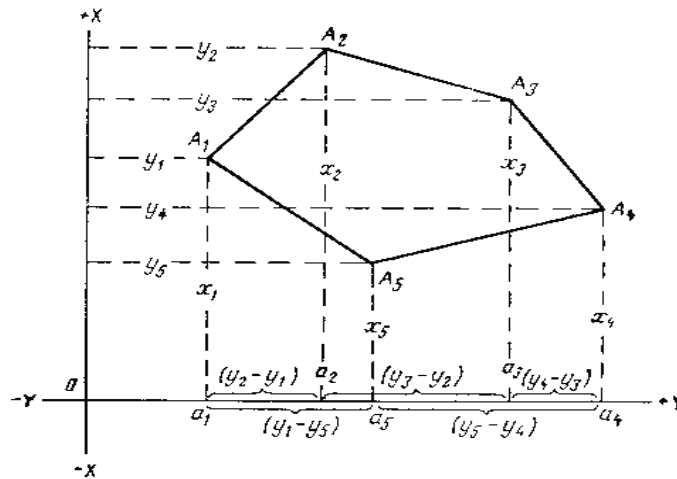
Справді,  $S_{A_1 A_2 A_3} = S_{y_1 A_1 A_2 y_2} + S_{y_2 A_2 A_3 y_3} - S_{y_1 A_1 A_3 y_3}$ , а наприклад, площа трапеції  $y_1 A_1 A_3 y_3$  дорівнює  $S = (y_2 - y_1)(x_1 + x_2)/2$ .

Значимо, площа фігури, заданої точками  $A_1, \dots, A_n, n \geq 4$ , залежить від порядку, в якому задано вершини. Наприклад, якщо вершини розміщені на площині так, як показано на малюнку, то дістанемо фігуру, яка називається чотиривершинником (його площа заштрихована).



3) Структуру формули можна зрозуміти на прикладі многокутника.

Нехай потрібно визначити площу многокутника  $A_1, A_2, A_3, A_4, A_5$  з координатами вершин  $x_1, y_1; x_2, y_2; x_3, y_3; x_4, y_4; x_5, y_5$ . Площа многокутника  $S$  можна представити трапеціями, у яких абсциси є основними, а різниця ординат сусідніх точок висотами.



$$S = a_1 A_1 A_2 a_2 + a_2 A_2 A_3 a_3 + a_3 A_3 A_4 a_4 - a_5 A_5 A_4 a_4 - a_1 A_1 A_5 a_5.$$

$$2S = (x_1 + x_2)(y_2 - y_1) + (x_2 + x_3)(y_3 - y_2) + (x_3 + x_4)(y_4 - y_3) + (x_4 + x_5)(y_5 - y_4) + (x_5 + x_1)(y_1 - y_5).$$

Після розкриття дужок і приведення подібних членів отримаємо

$$2S = x_1 y_2 - x_2 y_1 + x_2 y_3 - x_3 y_2 + x_3 y_4 - x_4 y_3 + x_4 y_5 - x_5 y_4 + x_5 y_1 - x_1 y_5$$

Після винесення за дужки  $x_1, x_2, x_3, x_4, x_5$  будемо мати

$$2S = x_1(y_2 - y_5) + x_2(y_3 - y_1) + x_3(y_4 - y_2) + x_4(y_5 - y_3) + x_5(y_1 - y_4)$$

а якщо з винести за дужки  $y_1, y_2, y_3, y_4, y_5$  то будемо мати

$$2S = y_1(x_5 - x_2) + y_2(x_1 - x_3) + y_3(x_2 - x_4) + y_4(x_3 - x_5) + y_5(x_4 - x_1).$$

В скороченому вигляді ці формули можна записати так:

$$S = \frac{1}{2} \sum x_k (y_{k+1} - y_{k-1})$$

$$S = \frac{1}{2} \sum y_k (x_{k+1} - x_{k-1})$$

Після перетворення отримаємо формулу в її нормальному вигляді.

$$S = \frac{1}{2} \left| \sum_{k=1}^n (x_k + x_{k+1})(y_k - y_{k+1}) \right|, \text{ де } X_0, Y_0 = X_{n+1}, Y_{n+1}$$

У програмі використовуються змінні:

$n$  – кількість вершин;

$s$  – площа многокутника;

$x[1..n+1], y[1..n+1]$  – масиви координати вершин;

$i$  – змінна циклу.

На  $n+1$  позицію заносимо координати першої вершини.

Програмний код:

**Program Prog 3;**

var x,y:array[1..100] of real;

i,n:integer;

s:real;

begin

write('n=');

readln(n);

for i:=1 to n do



```

    readln(x[i],y[i]);
x[n+1]:=x[1];
y[n+1]:=y[1];
s:=0;
for i:=1 to n do
    s:=s+(x[i]*y[i+1]-x[i+1]*y[i]);
s:=1/2*abs(s);
writeln('s=',s:2:2);
end.

```

## 2. Визначення положення точки відносно

### n-кутника

#### Задача 1. Перевірка належності точки прямій

Підставляємо координати точки в рівняння прямої і дивимося, чи є вони рішенням даних рівнянь. Так - належить, Ні - не належить.

Нехай  $(x_1, y_1)$ ,  $(x_2, y_2)$  – точки на прямій, то рівняння прямої буде наступним:  $(x-x_1)(y_2-y_1)=(y-y_1)(x_2-x_1)$ .

Функція перевірки належності точки з координатами  $(x, y)$ , буде наступна:

```

Function ft(x,y:real):Boolean;
Begin
If (x-x1)*(y2-y1)=(y- y1)*(x2-x1) then ft:=true else ft:=false;
End;

```

#### Задача 2. Перевірка належності точки відрізка

Як визначити, чи належить точка  $A(x, y)$  відрізка з кінцевими точками  $B(x_1, y_1)$  і  $C(x_2, y_2)$ ?

- 1) Це можна зробити через довжини відрізків  $BA+AC=BC$ .
- 2) За належністю точки прямій, яка містить відрізок і розміщення точки між початком та кінцем відрізка.

Нехай  $(x_1, y_1)$ ,  $(x_2, y_2)$  – координати початку і кінця відрізка, то рівняння прямої буде наступним:

$$(x-x_1)(y_2-y_1)=(y-y_1)(x_2-x_1).$$

Функція перевірки належності точки з координатами  $(x, y)$ , буде наступна:

```

Function ft(x,y:real):Boolean;
Begin
If (x-x1)*(y2-y1)=(y- y1)*(x2-x1) then
If (((x>=x1)and(x<=x2)and(x1<=x2))or((x>=x2)and(x<=x1)and(x2<=x1)))
then ft:=true else ft:=false;
End;

```

3) Точки відрізка z можна описати рівнянням

$$pOB+(1-p)OC=z, 0\leq p\leq 1, OB \text{ і } OC - \text{ вектори.}$$

Якщо існує таке p,  $0\leq p\leq 1$ , що

$$pOB+(1-p)OC=A, \text{ то } A \text{ лежить на відрізку, інакше - ні.}$$

Рівність розписується по координатний так:

$$px_1+(1-p)x_2=x$$

$$py_1+(1-p)y_2=y$$

З першого рівняння знаходимо p, підставляємо в друге: якщо одержуємо рівність і  $0\leq p\leq 1$ , то A на відрізку, інакше - ні.

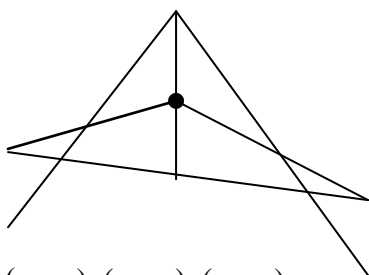
$$px_1+x_2-px_2=x \Rightarrow p(x_1-x_2)=x-x_2 \Rightarrow p= (x-x_2)/(x_1-x_2)$$

$$(x-x_2)/(x_1-x_2)y_1+(1-(x-x_2)/(x_1-x_2))y_2=y$$

*Програмний код і тестування проробить саостійно.*

### Задача 3. Перевірка належності точки трикутнику

Для трикутника можна розробити алгоритм, який обчислює площі трикутників утворених даною точкою і сторонами трикутника, якщо сума утворених трикутників рівна сумі трикутника, то точка належить трикутнику, інакше не належить.



Нехай  $(x_1,y_1), (x_2,y_2), (x_3,y_3)$  – координати вершин трикутника;

$(x,y)$  – координати точки для перевірки.

Програмний код:

**Program pr4;**

```
var x1,y1,x2,y2,x3,y3,x,y,a1,b1,c1,s1,s2,s3,st:real;
```

```
procedure line(xp,yp,xk,yk:real; var l:real);
```

```
begin
```

```
l:=sqrt(sqrt(xp-xk)+sqrt(yp-yk));
```

```
end;
```

```
procedure pl(a,b,c:real; var s:real);
```

```
var p:real;
```

```
begin
```

```
p:=(a+b+c)/2;
```

```
s:=sqrt(p*(p-a)*(p-b)*(p-c));
```

```

end;

begin
writeln('Введіть координати вершин трикутника');
readln(x1,y1,x2,y2,x3,y3);
writeln('Введіть координати точки');
readln(x,y);
line(x1,y1,x2,y2,a1); line(x1,y1,x,y,b1); line(x2,y2,x,y,c1);
pl(a1,b1,c1,s1);

line(x2,y2,x3,y3,a1); line(x2,y2,x,y,b1); line(x3,y3,x,y,c1);
pl(a1,b1,c1,s2);

line(x3,y3,x1,y1,a1); line(x3,y3,x,y,b1); line(x1,y1,x,y,c1);
pl(a1,b1,c1,s3);

line(x1,y1,x2,y2,a1); line(x2,y2,x3,y3,b1); line(x3,y3,x1,y1,c1);
pl(a1,b1,c1,st);

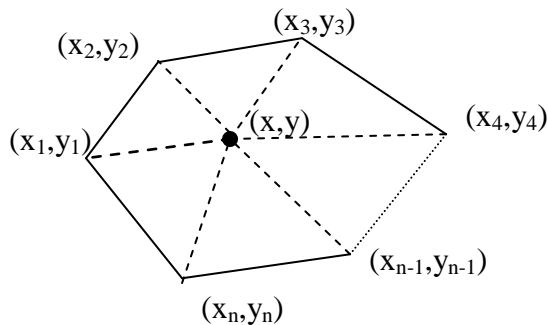
if s1+s2+s3=st then writeln('належить') else writeln('не належить');
end.

```

### Задача 3. Перевірка належності точки многокутнику

Під належністю точки многокутнику приймається належність точки внутрішній області многокутника або його межі. В тому числі, всі вершини належать многокутнику.

1) Аналогічно до задачі про належність точки трикутнику, можна розв'язати задачу для довільного опуклого многокутника, розбивши його на трикутники.



Програмний код:

```

Program Prog 5;
var xx,yy:array[1..100] of real;
x,y,a1,b1,c1,s1,s_mn, s_tr:real;
i,n:integer;
procedure line(xp,yp,xk,yk:real;var l:real);
begin
l:=sqrt(sqr(xp-xk)+sqr(yp-yk));
end;

procedure pl(a,b,c:real; var s:real);
var p:real;

```

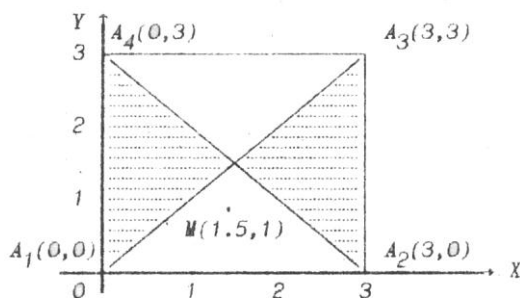
```

begin
p:=(a+b+c)/2;
s:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
writeln('Введіть кількість вершин многокутника');
readln(n);
writeln('Введіть координати вершин многокутника');
for i:=1 to n do readln(xx[i],yy[i]);
writeln('Введіть координати точки');
readln(x,y);
s_mn:=0;
for i:=1 to n-2 do begin
line(xx[i],yy[i],xx[i+1],yy[i+1],a1);
line(xx[i+1],yy[i+1],xx[i+2],yy[i+2],b1);
line(xx[i+2],yy[i+2],xx[i],yy[i],c1);
pl(a1,b1,c1,s1);
s_mn:=s_mn+s1;
end;
s_tr:=0;
xx[n+1]:=xx[1];
yy[n+1]:=yy[1];
for i:=1 to n do begin
line(xx[i],yy[i],xx[i+1],yy[i+1],a1);
line(xx[i],yy[i],x,y,b1);
line(xx[i+1],yy[i+1],x,y,c1);
pl(a1,b1,c1,s1);
s_mn:=s_mn+s1;
end;
if s_mn=s_tr then writeln('належить') else writeln('не належить');
end.

```

2) Для визначення положення точки відносно многокутника перевіряється той факт, що точка лежить по одну сторону від всіх ребер полігону. Якщо це не так, то вона зовні. В загальному випадку такий спосіб не працює.



Програмний код:

```

Program Prog 5;
var x,y:array[1..100] of real;
x0,y0:real;
i,n:integer;
b:Boolean;

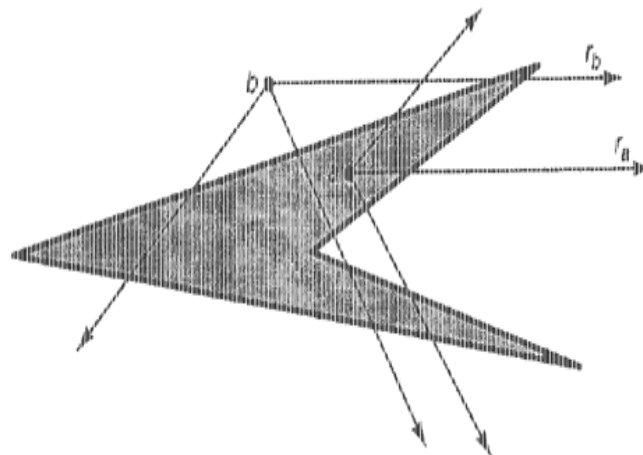
```

```

begin
writeln('Введіть кількість вершин многокутника');
readln(n);
writeln('Введіть координати вершин многокутника');
for i:=1 to n do readln(x[i],y[i]);
writeln('Введіть координати точки');
readln(x0,y0);
x[n+1]:=x[1];
y[n+1]:=y[1];
b:=true;
for i:=1 to n do begin
if (y0>y[i])=(y0<=y[i+1]) then
if x0-x[i]<(y0-y[i])*(x[i+1]-x[i])/(y[i+1]-y[i]) then b:=not(b);
if not(b) then writeln('належить') else writeln('не належить');
end.

```

3) Припустимо, що нам необхідно визначити належність точки  $A$  многокутнику  $p$ . Для цього з деякої віддаленої точки проведемо пряму лінію в точку  $A$ . На цьому шляху може зустрітися нуль або декілька перетинів сторін межі многокутника: при першому перетині ми входимо всередину многокутника, при другому — виходимо з нього, при третьому перетині знову входимо всередину і так до тих пір, поки не досягнемо точки  $A$ . Таким чином кожний непарний перетин означає попадання всередину многокутника  $p$ , а кожне парне — вихід з нього. Якщо ми потрапляємо в точку  $a$  з непарним числом перетинів сторін многокутника, то точка  $a$  лежить усередині полігону, а якщо виходить парне число перетинів, то точка знаходиться зовні многокутника  $p$ . Наприклад, промінь  $ra$  перетинає межу тільки одного разу, оскільки одиниця число непарне, то точка  $a$  знаходиться усередині полігону. Щодо точки  $b$  ми прийдемо до висновку, що вона лежить зовні многокутника, оскільки промінь  $rb$  перетинає сторони парне число разів (двічі).

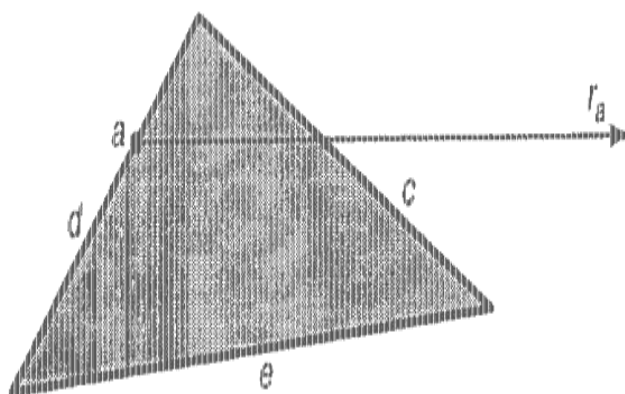


Побудова алгоритму на основі цієї ідеї базується на двох особливостях.

По-перше, для вирішення задачі підходить будь-який промінь, що починається в аналізованій точці  $A$ . Тому для простоти виберемо правий горизонтальний промінь  $ra$ , що починається в точці  $A$  і напрямлений вправо паралельно додатній напівосі  $x$ .

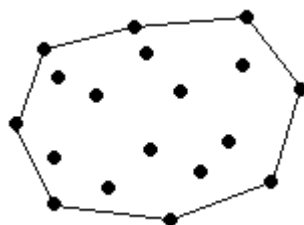
По-друге, порядок розташування ребер, що перетинаються променем  $ga$  байдужий, має значення лише парність (парна або непарна кількість) їх загального числа. Отже, замість моделювання руху уздовж променя  $ga$  для алгоритму достатньо визначити всі перетини ребер у будь-якому порядку, визначаючи парність у міру просування. Найпростішим розв'язком буде обхід межі полігону з перемиканням біта парності при кожному виявленні ребра, що перетинається променем  $ga$ .

Щодо горизонтального променя  $ga$ , направлено у право, розрізнятимемо три типи ребер многокутника: дотичне ребро, що містить точку  $A$ ; перетинаюче ребро, що не містить точку  $A$ ; байдуже ребро, яке зовсім не має перетин з променем  $ga$ . Наприклад, ребро  $c$  є перетинаючим ребром, ребро  $d$  — дотичним, а ребро  $e$  — байдужим.



### 3. Визначення опуклої оболонки

Нехай задано деяка множина точок на площині. Опукла оболонка - найменший випуклий многокутник, який містить дані точки.



Уявимо, що площина - це дерев'яна дошка з гвіздками в кожній точці з даної множини. Тепер натягнемо навколо гвіздків резинове кільце. Воно стягнеться і утворить опуклу оболонку, як показано на малюнку вище.

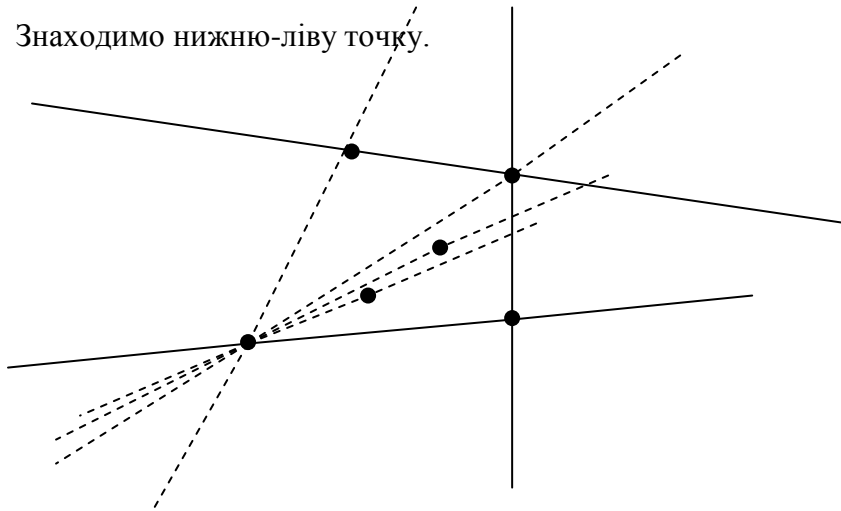
Алгоритм містить наступні операції:

- визначення лівої нижньої точки;
- належність всіх точок одній півплощині відносно прямої проведеної через дві точки;

- визначення точок опуклої оболонки.

Цей алгоритм обводить точки лінією, «завертаючи» в опуклу оболонку. На малюнку дана детальна ілюстрація його дії.

Знаходимо нижню-ліву точку.



Проводимо прямі через дану точку і всі інші точки. Перевіряємо належність всіх точок одній півплощині відносно даної прямої. Якщо це справджується, то знайдена точка є вершиною многокутника і з неї шукаємо аналогічно наступну точку.

В масиві T під номером MIN відмічаємо, знайдену точку як 1.

I координати занести в масиви XN,YN під номером 1.

Повторюємо поки пошук наступної точки дає результат :

- Для кожної точки крім відмічених точок в масиві T[i], шукаємо наступну точку, провівши через яку пряму всі інші точки лежать в одній півплощині;

- для цього перевіряємо, чи є хоча б одна точка лежить в правій площині, якщо немає то дана точка є вершиною опуклої оболонки;

- якщо є точки, то перевіряємо аналогічно для лівої півплощини;

- знайдену точку заносимо в масиви XN,YN під номером k+1;

- в масиві T під номером I відмічаємо, знайдену точку як k+1;

- відмічаємо дану точку як першу і шукаємо наступну;

Виводимо послідовно точки з XN,YN.

Програмний код:

```
PROGRAM OBOLONKA;  
  VAR X,Y,XN,YN,T:ARRAY[1..100] OF INTEGER;  
      N,I,J,K,MIN,MAX,LICH:INTEGER;  
      X1,Y1,X2,Y2,XMIN,XMAX,YMIN:INTEGER;  
      POISK,PT:BOOLEAN;
```

```
BEGIN
```

```

WRITE('N=');
readln(n);
FOR I:=1 TO N DO READLN(X[I],Y[I]);
FOR I:=1 TO N DO T[I]:=0;

MIN:=1;
FOR I:=2 TO N DO IF X[I]<X[MIN] THEN MIN:=I;
XMIN:=X[MIN];

K:=1;
X1:=X[MIN]; Y1:=Y[MIN];
XN[K]:=X1; YN[K]:=Y1;
T[MIN]:=K;

POISK:=TRUE;
WHILE POISK DO BEGIN
  POISK:=FALSE;
  FOR I:=1 TO N DO
    IF T[I]=0 THEN BEGIN
      X2:=X[I];
      Y2:=Y[I];
      PT:=TRUE;
      {}
      FOR J:=1 TO N DO
        IF (X[J]-X1)*(Y2-Y1)-(Y[J]-Y1)*(X2-X1)<0 THEN
          PT:=FALSE;

      IF NOT (PT) THEN
        FOR J:=1 TO N DO
          IF (X[J]-X1)*(Y2-Y1)-(Y[J]-Y1)*(X2-X1)>0 THEN
            PT:=FALSE;

      IF PT THEN BEGIN K:=K+1;
        X1:=X2;
        Y1:=Y2;
        T[I]:=K;
        XN[K]:=X1;
        YN[K]:=Y1;
        POISK:=TRUE;
      END;
    END;
  END;
END;
WRITELN;
FOR I:=1 TO K DO WRITELN(XN[I],',',YN[I]);
END.

```

## 5. Приклади задач на обчислювальну геометрію

### Задача 1.

На площині задано дві точки  $A(x_1, y_1)$  і  $B(x_2, y_2)$ . Визначити, який з відрізків -  $OA$  або  $OB$  утворює більший кут з віссю  $Ox$ .

### Задача 2.

Чи належить точка площини  $A$  відрізку з кінцевими точками  $B$  і  $C$ ?

### Задача 3.

1. Опуклий багатокутник задається координатами вершин при його обході по годинниковій або проти годинникової стрілки. Контур багатокутника не має самоперетинів. Визначити напрям обходу.



2. Виконати те ж саме, але тільки у разі неопуклого многокутника.

---

**Задача 4.**

Відрізок на площині задається двома не співпадаючими кінцевими точками  $X(x_1, x_2)$  і  $Y(y_1, y_2)$ . З точки  $Z(z_1, z_2)$  до прямої, що містить відрізок  $[X, Y]$ , проводиться перпендикуляр  $P$ . Визначити, чи потрапляє перпендикуляр  $P$  на відрізок  $[X, Y]$  або на його продовження.

---

**Задача 5.**

Многокутник на площині задається координатами своїх  $N$  вершин в порядку обходу їх по контуру за годинниковою стрілкою. Вважається, що контур самоперетинів не має. Знайти площу, периметр і кути многокутника.

---

**Задача 6.**

Визначити, чи перетинається пряма  $ax + b = y$  і відрізок з кінцями  $(x_1, y_1)$   $(x_2, y_2)$ .

---

**Задача 7.**

Відрізки на площині задаються парами цілочисельних координат кінцевих точок. Визначити, чи перетинаються 2 відрізки.

---

**Задача 8.**

1. Трикутник на площині задається цілочисельними координатами вершин. Для заданої точки  $Z(x, y)$  визначити, чи належить вона стороні трикутника або лежить всередині або зовні нього.
  2. Многокутник на площині задається координатами своїх  $N$  вершин в порядку обходу їх по контуру за годинниковою стрілкою (контур самоперетинів не має). Для заданої точки  $Z(x, y)$  визначити, чи належить вона стороні многокутника або лежить всередині або зовні нього.
- 

**Задача 9.**

На площині задані  $n$  відрізків координатами кінцевих точок. Кінці відрізків задаються двома парами координат  $(x_1[i], y_1[i])$ ,  $(x_2[i], y_2[i])$ ,  $1 \leq i \leq n$  (кінці належать відрізку). Необхідно знайти пряму, що має загальні точки з максимальним числом відрізків, і надрукувати в порядку зростання номера тих відрізків, які ця пряма перетинає.

---

**Задача 10.**

$N$  точок на площині задані своїми координатами. Знайти такий мінімальний за площею опуклий многокутник, що всі  $N$  точок лежать або усередині цього многокутника, або на його межі (такий опуклий многокутник називається опуклою оболонкою).

---

**Задача 11.**

На ґратах розміру  $m \times n$  задані  $K$  точок своїми координатами. Необхідно визначити, чи можна побудувати опуклий многокутник такий, що кожна точка належить деякій стороні.

---

**Задача 12.**

$N$  точок на площині задані своїми координатами. Знайти порядок, в якому можна з'єднати ці точки, щоб вийшов  $N$ -кутник (тобто не було б перетинів сторін).

---

**Задача 13.**

Уявіть собі, що в зошиті Ви замалювали на листі якусь кількість клітинок і отримали клітинну фігуру. Скільки осей симетрії має задана клітинна фігура.

---

Пояснення :

1) Задається  $S$  - число тестів. Для кожного тесту задаються  $NI$  розмір фігури по вертикалі,  $NJ$  - розмір фігури по горизонталі ( $NI < 101$ ,  $NJ < 81$ ) і сама фігура у вигляді  $NI$  рядків з пропусків і одиниць по  $NJ$  символів в кожному рядку.

2) Числа  $S$ ,  $NI$ ,  $NJ$  займають при введенні по три позиції.

Приклад .

Вхідні дані :

2 ( кількість тестів )

2 ( розмір 1-ої фігури по вертикалі )

4 ( розмір 1-ої фігури по горизонталі )

1111

1\_\_1

3 ( розмір 2-ої фігури по вертикалі )

5 ( розмір 2-ої фігури по горизонталі )

11111

111\_\_

111\_\_

Вихідні повідомлення:

1-А ФІГУРА МАЄ 1 ОСЕЙ СИМЕТРІЇ

2-А ФІГУРА МАЄ 0 ОСЕЙ СИМЕТРІЇ

---

#### Задача 14.

Прямокутник  $ABCD$  заданий координатами своїх вершин. На протилежних сторонах  $AB$  і  $CD$  задані послідовності  $R1$  і  $R2$  з  $N$  точок розбиття, а на сторонах  $BC$  і  $AD$  –  $R3$  і  $R4$  з  $M$  точок розбиття. Нумерація елементів послідовності  $R1$  і  $R2$  починається відповідно від точок  $A$  і  $D$ , а в  $R3$  і  $R4$  -- від  $B$  і  $A$ . З'єднавши відрізками точки з однаковими номерами в розбитті  $R1$  і  $R2$ , а потім в розбитті  $R3$  і  $R4$ , отримаємо розбиття  $Q$  прямокутника  $ABCD$  на безліч чотирикутників. Побудувати алгоритм, який визначить чотирикутник розбиття  $Q$  з найбільшою площею, за умови, що відрізки, які сполучають точки розбиття  $R1$  і  $R2$  паралелі стороні  $AD$ . Послідовності  $R1$ ,  $R2$ ,  $R3$  і  $R4$  задаються як масиви з довжин відрізків розбиття відповідних сторін прямокутника.

---

#### Задача 15.

На прямій задано  $N$  точок з координатами  $X_1, X_2, \dots, X_n$ . Написати програму, яка знаходить на прямій таку точку  $z$ , сума відстаней від якої до даних  $N$  точок мінімальна.

---

#### Задача 16.

На двовимірній площині задано  $N$  точок з координатами  $(X_1, Y_1)$ ,  $(X_2, Y_2)$  ...  $(X_n, Y_n)$ . Написати програму, яка знаходить таку точку  $Z(x, y)$ , сума відстаней від якої до інших мінімальна і:

a)  $Z$  - одна із заданих точок;

b)  $Z$  - довільна точка площини.

---

#### Задача 17.

На площині розташовані  $N$  точок, задані своїми координатами. Знайти на осі абсцис точку, найбільша з відстаней від якої до вибраних точок була б мінімальним.

---

#### Задача 18.

На двовимірній площині задано  $N$  точок з координатами  $(X_1, Y_1)$ ,  $(X_2, Y_2)$  ...  $(X_n, Y_n)$ . Написати програму, яка з цих точок виділяє вершини квадрата, що містить максимальне число заданих точок.

ПРИМІТКА: передбачається, що точки, розташовані на сторонах квадрата, належать

йому.

---

### Задача 19.

Задана на площині множина з  $N$  прямокутників, сторони яких паралельні осям координат, при цьому кожний прямокутник задається координатами лівої нижньої і правої верхньої його вершин.

Скласти алгоритм визначення найбільшого натурального числа  $K$ , для якого існує точка площини, що належить одночасно  $K$  прямокутникам.

---

### Задача 20.

На квадратному торті  $N$  свічок. Чи можна одним прямолінійним розрізом розділити його на дві рівні за площею частини, одна з яких не містила б жодної свічки? Свічки вважатимемо точками, у яких відомі їх цілочисельні координати  $X[1], Y[1]; \dots; X[N], Y[N]$  (початок координат - в центрі торта); розріз не може проходити через свічку.

---

### Задача 21.

Дано  $N, N > 1$ , прямокутників, для яких передбачається, що:

А). Сторони будь-якого прямокутника паралелі координатним осям і прямокутник задається кінцями однієї з діагоналей.

В). Кожний прямокутник має загальні внутрішні точки з хоча б одним з інших і не має загальних вершин, сторін або частин сторін ні з одним з решти прямокутників.

Скласти програму, яка розв'язує наступні задачі:

1. За допомогою послідовності точок визначити зовнішній контур фігури  $F$ , що є об'єднанням прямокутників - ламану замкнуту криву. Приклад на кресленні.

2. Визначити чи містить фігура  $F$  "дірки", тобто замкнуті фігури, які їй не належать.

3. Розкласти фігуру на якнайменше можливе число прямокутників, які не перетинаються і можуть мати загальні сторони або частини сторін, а їх об'єднання дає фігуру  $F$ .

4. Обчислити периметр і площу фігури  $F$ .

Примітка.

1. Задачі 3,4 розв'язуються тільки для фігур, які не містять "дірки".

2. Повний розв'язок містить:

- аналіз (обґрунтування) розв'язку;

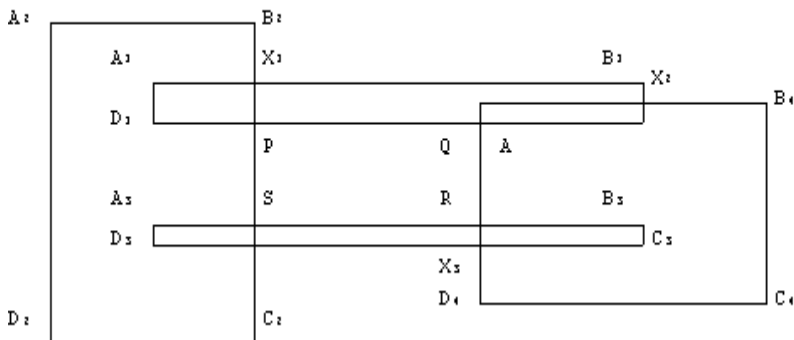
- текст програми з відповідними коментарями;

- виконання тестового прикладу, який буде даний при прийманні роботи.

Об'єднання прямокутників  $A_i B_i C_i D_i, i=1,2,3,4$  є

фігура  $F=A_2 B_2 X_1 B_1 X_2 B_4 C_4 D_4 X_3 X_4 C_2 D_2$

фігура  $F$  містить єдину "дірку" PQRS.

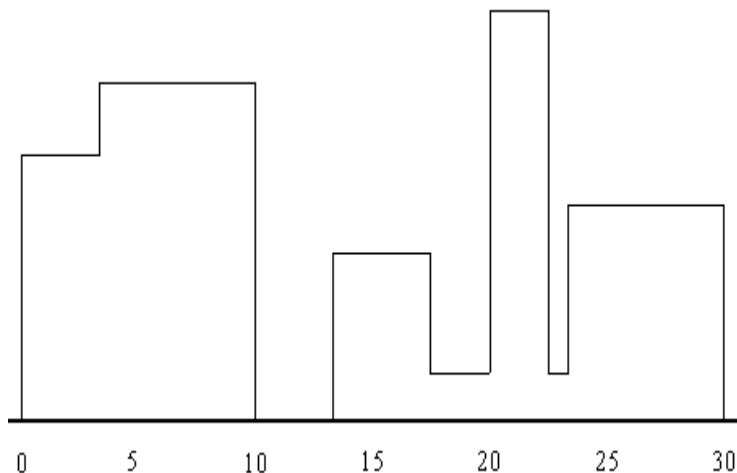


---

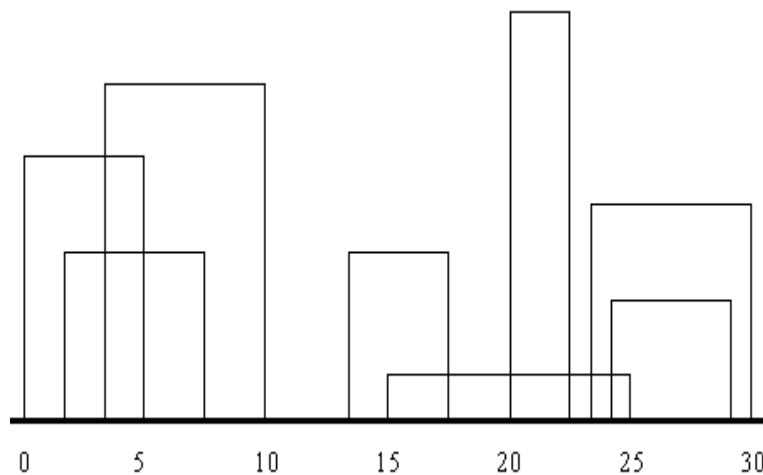
### Задача 22.

## Контур міста.

Необхідно написати програму - помічник архітектора в малюванні контура міста. Місто задається розташуванням будівель. Місто розглядається як двовимірний і всі будівлі в ньому - прямокутники, що мають однакову основу (місто побудовано на рівнині). Будівлі задаються трійкою чисел  $(L[i], H[i], R[i])$  де  $L[i]$  і  $R[i]$  є координати лівої і правої стін будівлі  $i$ , а  $H[i]$  - висота цієї будівлі. На малюнку 1 будівлі описуються трійками  $(1,11,5)$ ,  $(2,6,7)$ ,  $(3,13,9)$ ,  $(12,7,16)$ ,  $(14,3,25)$ ,  $(19,18,22)$ ,  $(23,13,29)$ ,  $(24,4,28)$  а контур, показаний на мал. 2, задається послідовністю  $(1,11,3,13,9,0,12,7,16,3,19,18,22,3,23,13,29,0)$  (про спосіб формуванні цієї послідовності див. нижче).



Мал. 1



Мал. 2

## Введення.

Введення є послідовністю трійок, які задають будівлі. Всі координати є цілі числа, менші 10000. У вхідному файлі мінімум одна і максимум 50 будівель. Кожна трійка, що позначає будівлю знаходиться в окремому рядку у вхідному файлі. Всі цілі числа в трійці розділені одним або декількома пропусками. Трійки відсортовані по  $L[i]$ , тобто по лівій x-координаті будівлі, таким чином, будівля з найменшою лівою x-координатою є першою у вхідному файлі.

## Виведення.

Виведення складатиметься з вектора, що описує контур, як показано в прикладі вище. У векторі контура  $(v[1], v[2], v[3] \dots, v[n-2], v[n-1], v[n])$ ,  $v[i]$ , коли  $i$ -парне число, означає горизонтальну лінію (висоту). Коли  $i$ -непарне,  $v[i]$ -означає вертикальну лінію (x-координату). Вектор контура визначатиме маршрут, що починається з мінімальної x-координати і проходить по всіх вертикальних і горизонтальних лініях, що визначають

контур. Останній елемент у векторі лінії контура буде 0.

---

### Задача 23.

Нижня ліва і верхня права вершини прямокутника  $A$  мають координати  $(0,0)$  і  $(V,W)$  відповідно. Множина  $S$  з  $N$  точок задається парами координат  $(x[i],y[i])$ ,  $1 \leq i \leq N$ .

Знайти такий прямокутник  $G$  максимальної площі, що його сторони паралелі сторонам  $A$ ,  $G$  повністю лежить в  $A$  ( $G$  і  $A$  можуть мати загальні граничні точки) і жодна точка з  $S$  не лежить усередині  $G$  (але може лежати на його стороні). Надрукувати величину площі  $G$  і координати нижньої лівої і верхньої правої вершин цього прямокутника.

Якщо таких прямокутників декілька, то вивести інформацію про кожного.

Зауваження: в множині  $S$  ніякі дві точки не лежать на одній прямій, паралельній стороні  $A$ .

Необхідно:

1. Організувати введення даних у вигляді

< Введіть  $V$  і  $W$  -- координату верхньої правої вершини прямокутника  $A$  >

< Введіть  $N$  -- число точок >

< Введіть координати точок >

$x[1] \rightarrow y[1] \rightarrow$

.....

$x[N] \rightarrow y[N] \rightarrow$

2. Вивести результати у вигляді

< Максимальна площа прямокутника = >

< Координати нижньої лівої і верхньої правої вершин прямокутника(ів) >

$x1 \rightarrow y1 \rightarrow x2 \rightarrow y2 \rightarrow$

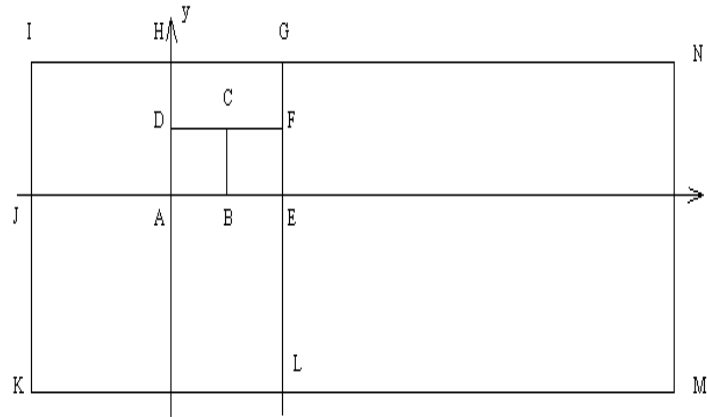
---

### Задача 24.

В першому квадранті координатної системи  $OXY$  намальований перший квадрат -  $ABCD$ , довжина сторони якого рівна 1 і вершина  $A$  знаходиться на початку координат. Потім намальовані: другий квадрат -  $BEFC$ , третій -  $DFGH$ , четвертий -  $JAHI$ , п'ятий -  $KLEJ$ , шостий -  $LMNG$ , і так далі 'по спіралі' (мал.1).

Написати програму, яка для введених цілих чисел  $x$  і  $y$  визначає і виводить номер квадрата, якому належить точка  $P(x,y)$ . Якщо точка  $P$  лежить на сторонах квадратів або у вершинах, то вважатимемо, що вона належить квадрату з якнайменшим номером з можливих.

Приклади:	X	Y	Результат
	2	1	1
	-1	0	4
	13	2	10



Мал.1

---

**Задача 25.**

На площині  $N$  різних точок задані своїми координатами. Знайти рівняння прямої, що ділить дані точки на 2 рівносильні підмножини (тобто на підмножини з однаковою кількістю елементів).

---

**Задача 26.**

Знайти перетин і об'єднання двох опуклих багатокутників. Многокутники задаються координатами вершин в порядку обходу по контуру.

---

**Задача 27.**

$N$ -кутник на площині задається координатами вершин в порядку обходу по контуру (контур самоперетинів не має). Для точки  $Z(x,y)$  знайти мінімальну відстань до контура  $N$ -кутника.

---

**Задача 28.**

На площині задано  $N$  точок своїми координатами і матриця  $C(N*N)$ ;  $C(i,j)=C(j,i)=1$  у випадку, якщо вершини  $i$  і  $j$  сполучені відрізком і 0 інакше. Відомо, що будь-яка вершина сполучена принаймні з двома іншими, і що відрізки перетинаються тільки в кінцевих точках. Таким чином, вся площина розбивається на безліч багатокутників. Задана точка  $Z(x,y)$ . Знайти мінімальний за площею багатокутник, який містить  $Z$ , або видати повідомлення, що такого не існує. Якщо  $Z$  належить якомусь відрізку, то видати його кінцеві точки, якщо  $Z$  лежить в багатокутнику, то видати його вершини в порядку обходу по контуру.

---

**Задача 29.**

Називатимемо два багатокутники подібними, якщо існує взаємно-однозначне відображення сторін цих двох фігур таке, що відповідні сторони пропорційні з коефіцієнтом пропорційності  $K$ , а кути, утворені двома відповідними сторонами, рівні. Визначити, чи подібні два багатокутники. Многокутники задаються на площині координатами вершин контурів. Вершини в контурі перераховуються в порядку обходу проти годинникової стрілки.

Примітка: оскільки всі обчислення на ЕОМ проводяться з обмеженою точністю, то вважати, що дві величини рівні якщо вони співпадають з точністю до двох знаків після коми.

ВВЕДЕННЯ: <з файлу T.TXT>

- <Кількість вершин в контурі:>  $N$
- <Многокутник 1:>

- <Координати вершини 1:>  $x_{11}, y_{11}$
- .....
- <Координати вершини N:>  $x_{1N}, y_{1N}$
- < Многокутник 2:>
- <Координати вершини 1:>  $x_{21}, y_{21}$
- .....
- <Координати вершини N:>  $x_{2N}, y_{2N}$

#### ВИВЕДЕННЯ:

< Многокутники неподібні > або

<Многокутники подібні з  $k \Rightarrow k$

#### Задача 30.

Задане натуральне  $N$  і дві послідовності цілих чисел  $(a_1, a_2, \dots, a_n)$  і  $(b_1, b_2, \dots, b_n)$ . Задано також два числа  $X_0$  і  $X_1$ ,  $X_0 < X_1$ .

1. Знайти числа  $t(0), t(1) \dots, t(p)$ ,  $p \leq N$ , такі, що  $X_0 = t(0) < t(1) < \dots < t(p) = X_1$  і вказати для кожного відрізка  $[t(j-1), t(j)]$ ,  $1 \leq j \leq p$ , таке число  $k$ ,  $1 \leq k \leq N$ , що для всіх  $i$ ,  $1 \leq i \leq N$ , і для всіх  $x$  з  $[t(j-1), t(j)]$  справедливо нерівність  $a_k * x + b_k \geq a_i * x + b_i$ .

2. Знайти числа  $s(0), s(1) \dots, s(Q)$ , такі, що  $X_0 = s(0) < s(1) < \dots < s(Q) = X_1$ , і вказати для кожного відрізка  $[s(j-1), s(j)]$ ,  $1 \leq j \leq Q$ , таку перестановку  $(i_1, i_2, \dots, i_N)$  чисел  $1, 2, 3, \dots, N$ , що для всіх  $x$  з  $[s(j-1), s(j)]$  справедливо нерівність

$$a_{i_1} * x + b_{i_1} \leq a_{i_2} * x + b_{i_2} \leq \dots \leq a_{i_N} * x + b_{i_N}$$

і для всіх відрізків відповідні перестановки різні.

#### Задача 31.

На площині задано безліч точок  $A$  і безліч прямих  $B$ . Знайти дві такі різні точки з  $A$ , що пряма паралельна, яка проходить через них найбільшій кількості прямих з  $B$ .

#### Задача 32.

В правильному  $n$ -кутнику провели декілька діагоналей, причому ніякі три не перетинаються в одній точці. На скільки частин діагоналі розбили  $n$ -кутник? Діагоналі задані номерами вершин  $n$ -кутника, які вони сполучають, всі вершини перенумеровані по порядку числами  $1, \dots, n$ .

#### Задача 33.

Круг розрізає несамоперетинаюча ламана, координати вершин якої задані парами натуральних чисел  $(x_1, y_1) \dots (x_k, y_k)$ . Перша і остання вершини лежать на межі круга, а інші усередині нього. Визначити, чи можна роз'єднати дві частини круга, що вийшли (вихід з площини і повороти частин, що рознімаються, не допускається).

#### Задача 34.

Серед трикутників з вершинами в заданій множині точок на площині вказати такий, сторони якого містять максимальне число точок заданої множини.

#### Задача 35.

Всі стіни будинку мають довжину 5м. Північна і південна сторони - білі, західна і східна - сині. Людина пройшла від південно-східного кута будинку  $A$  метрів на південь,  $B$  метрів на схід і  $C$  метрів на північ і подивився на будинок.

Написати алгоритм, який визначає, що бачить людина.

#### Задача 36.

На місцевості, що є ідеально рівною поверхнею, стоїть високий забір. План забору є замкнутою ламаною без самоперетинів. Ця ламана задається  $N$  парами координат своїх вершин в порядку обходу обмеженої забором області проти годинникової стрілки. Вершини пронумеровані від 1 до  $N$ ,  $N < 100$ .

В точці  $(x,y)$  стоїть людина ( $(x,y)$  не може лежати на ламаній). Вважаючи, що кожній ланці ламаної стає у відповідність пари номерів кінцевих вершин, вказати, які ланки людина побачить повністю або частково як невироджений відрізок, а які - взагалі ні. Якщо при погляді ланку видно як точку або як пару, точок, то вважаємо, що вона не видима.

---

### **Задача 37.**

На гранях двох рівних правильних тетраедрів  $N$  і  $M$  написані числа  $N_1, N_2, N_3, N_4$  і  $M_1, M_2, M_3, M_4$ .

Чи можна сумістити тетраедри так, щоб на співпадаючих гранях виявилися однакові числа?

## **Література**

1. Інформатика. Програми для загальноосвітніх навчальних закладів. – Запоріжжя: Прем'єр, 2003. – 304 с.
2. Баратків А.Б., Гринчишин Я.Т., Ломакович А.М., Рамський Ю.С. Turbo Pascal: Алгоритми і програми: Чисельні методи в фізиці та математиці: Навч. посібник –К.:Вища шк.,1992.- 247с.
3. Грузман М.З. Эвристика в информатике. – Винница: Арбат, 1998.-308с.
4. Пономарев В.И. Учебное пособие по программированию на языке Turbo Pascal. Пособие предназначено для обучения 8-11 классов основам информатики и вычислительной техники. Симферополь. Крымское учебно-педагогическое государственное издательство, 1998,- 256 с.
5. Фаронов В.В. Турбо Паскаль 7.0. Навчальний курс. Учебное пособие. Издание 7-е, переработаное.- М.: «Нолидж», 2000.-576 с.
6. А.Х.Шень Программирование. Теоремы и задачи.
7. <http://algotlist.manual.ru/> - широкий огляд алгоритмів і методів розв'язування задач.