Алгоритми в геометрията

# СЪДЪРЖАНИЕ

## ВЪВЕДЕНИЕ

За решаването на някои геометрични задачи (метрични,от изпъкналост, екстремални) може успешно да се използува компютър. Получените резултати могат да се онагледят с чертежи и може бързо да се даде отговор на задачи, които са нерешими в реално време без компютър.

В темата са включени конкурсни геометрични задачи,давани на състезания по информатика и публикувани в сп."Математика", "Обучението по математика и информатика", "Компютър за вас" , "Компютър". Решенията са реализирани на езика Pascal. Голяма част от условията на задачите са взети от [3].

Най-напред се въвеждат необходимите геометрични елементи - декартова координатна система, множества от точки, координати на точка, уравнение на права в равнината, взаимно положение на две прави, разстояние от точка до права, ъгъл между две прави. Следва въвеждане на необходимия метричен елемент (дължина на отсечка, лице на триъгълник, лице на многоъгълник) и на комбинаторните обекти - пермутации и комбинации без повторение.

Задачите, които се разглеждат в темата са разнообразни и за решаването на повечето от тях се изисква използването, и на комбинаторен елемент, което допринася за развитие на мисленето и разширяване познанията на учениците. Темата би могла да се използува за извънкласна работа по информатика в математически гимназии.

В началото на темата се прилагат най-често използуваните в задачите подпрограми:

*ПП за въвеждане на цяло число;
*ПП за въвеждане координатите на N точки в равнината;
*Дължина на отсечка;
*Лице на триъгълник;
*Лице на многоъгълник;
*ПП за намиране минимум и максимум;
*Проверка за изпъкналост на многоъгълник;
*ПП за генериране на пермутации;
*ПП за генериране на комбинации;
*Реализация на N вложени цикъла;

Задачите са групирани така:

**1.Задачи от изпъкналост** на фигури, за решаването на които се използува разгледания геометричен елемент.

**2.Метрични задачи** - за намиране дължина на отсечка, ъгъл между две прави, периметър и лице на многоъгълници, за решаване на някои от които се използува и комбинаторен елемент.

**3.Общи задачи** свързани с:

- прави;
- окръжности;
- многоъгълници;
- задачи, в които участвуват различни геометрични фигури;
- пространствени задачи;
- задачи, свързани с множества от точки и операции с множества.

В края на темата са приложени **конкурсни геометрични задачи**. Част от задачите, които са основни и необходими за решаването на други задачи в темата са решени, а останалите са дадени с упътване или за самостоятелна работа.

В края на работата е предложен примерен тематичен план за школа по информатика в рамките на 180 учебни часа. При работа с изявени ученици могат да се използват и отделни теми, които могат да спомогнат за цялостната им подготовка в извънкласната работа.

# 1.Теоретични сведения.

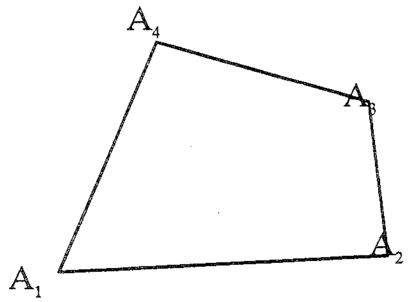
## 1.1 Основни понятия.Геометричен елемент.

Разглежданията ще извършваме в равнината спрямо декартова координатна система Oxy. Ще въведем някои основни понятия.[6]

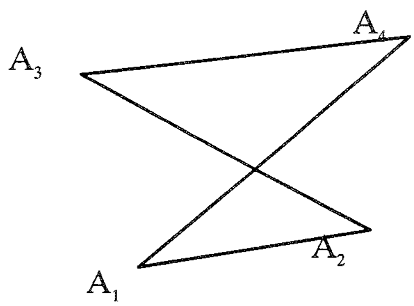
**Определение за многоъгълник**

Всяка затворена начупена линия с краен брой звена се нарича **многоъгълник.**

Един многоъгълник се нарича **прост**, ако никои две несъседни страни нямат обща точка.



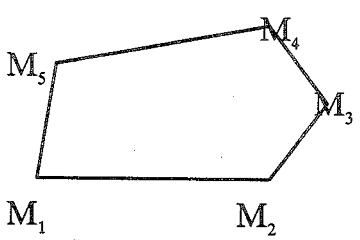

**прост многоъгълник**



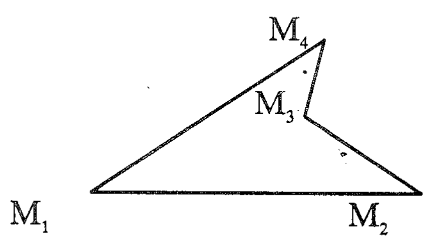

**не е прост**

Един многоъгълник е **изпъкнал**, ако заедно с всеки две точки съдържа и отсечката, която ги съединява. Друго еквивалентно определение, което се използва за проверка за изпъкналост е следното:

**Многоъгълникът М се нарича изпъкнал**, ако за всеки негов връх съществува съседен, така че останалите върхове на многоъгълника са от една и съща страна на правата, която свързва тези два върха.



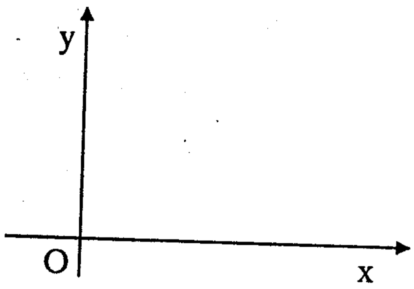

**изпъкнал многоъгълник**



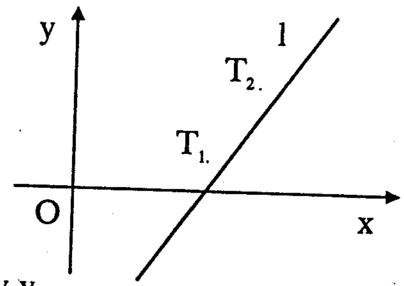

**не е изпъкнал**

### 1.1.1 Уравнение на права в равнината.

Нека в равнината е дадена правоъгълна координатна система Oxy.




Нека спрямо Oxy $T_1\ (x_1,y_1)$, $T_2(x_2,y_2)$
Как изглежда уравнението на на правата l през точките $T_1$ и $T_2$ ?




$$l:\ \frac{x-x_1}{x_2-x_1}=\frac{y-y_1}{y_2-y_1}$$

$F(x,y)=\ (x-x_1)(y_2-y_1)-(y-y_1)(x_2-x_1)$
$(x-x_1)(y_2-y_1)-(y-y_1)(x_2-x_1)=0$
$(y_2-y_1)x-x_1(y_2-y_1)-y(x_2-x_1)+y_1(x_2-x_1)=0$ $(y_2-y_1)x+(x_1-x_2)y+y_1(x_2-x_1)-x_1(y_2-y_1)=0$
$(y_2-y_1)x+(x_2-x_1)y+\ y_1(x_2-x_1)-\ x_1(y_2-y_1)=0$

Полагаме $A=\ y_2-y_1$
$B=\ \cancel{x_2-x_1}=x_1-x_2$
$C=y_1(x_2-x_1)-x_1(y_2-y_1)$

и получаваме уравнението на правата $l: Ax+By+C=0$ , $A,B,C\in R$ .
Точката $(p,q)\in l\ \Leftrightarrow\ F(p,q)=0$ .

### 1.1.2 Взаимно положение на права и две точки.

Правата l разделя равнината на две полуравнини. Едната от тях приемаме за положителна, а другата за отрицателна. Нека е положителна полуравнината в която $F(x,y)>0$. В сила е следното твърдение:

$F(x',y').F(x'',y'')<0 \Leftrightarrow$ когато точките с координати $(x',y')$ и $(x'',y'')$ са от различни страни на правата с уравнение $F(x,y)=0$.

$F(x',y').F(x'',y'')>0 \Leftrightarrow$ когато точките $(x',y')$ и $(x'',y'')$ са от една и съща страна на правата с уравнение $F(x,y)=0$.

### 1.1.3 Взаимно положение на две прави.

Нека правите $l_1$ и $l_2$ имат следните общи уравнения:

$$\left| \begin{array}{l} l_1: a_1x+b_1y+c_1=0 \\ l_2: a_2x+b_2y+c_2=0 \end{array} \right.$$

$a,b,c \in R$

Правите $l_1$ и $l_2$ се пресичат ако системата (1) има решение т.е. ако

$$D=\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} \qquad D_1=\begin{vmatrix} -c_1 & b_1 \\ -c_2 & b_2 \end{vmatrix} \qquad D_2=\begin{vmatrix} a_1 & -c_1 \\ a_2 & -c_2 \end{vmatrix}$$

$$x=\frac{D_1}{D} \qquad y=\frac{D_2}{D}$$

Решението на системата е:

$$\left| \begin{array}{l} x=\dfrac{c_1b_2-c_2b_1}{b_1a_2-b_2a_1} \\ y=\dfrac{a_1c_2-a_2c_1}{b_1a_2-b_2a_1} \end{array} \right.$$

ако $b_1a_2-b_2a_1 \neq 0$. Нека $k=\dfrac{a}{b}$.

Правите са успоредни, когато $k_1=k_2$.

Правите се сливат, когато $\dfrac{a_1}{a_2}=\dfrac{b_1}{b_2}=\dfrac{c_1}{c_2}$.

правите са перпендикулярни, когато $k_1.k_2=-1$.

### 1.1.4 Ъгъл между две прави.

Нека уравненията на двете прави са:

$y=k_1x+b_1$ - уравнение на права с ъглов коефициент k и отрез от

$y=k_2\,x+b_2$ ординатната ос в.

Нека $\varphi$ е острият ъгъл между правите. Тогава

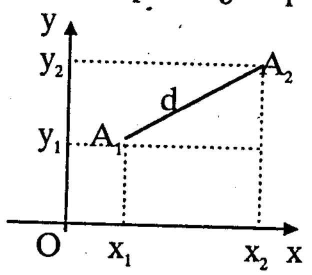
$$tg\,\varphi = \left| \frac{k_2-k_1}{1+k_1k_2} \right| \qquad \varphi = arctg \left| \frac{k_2-k_1}{1+k_1k_2} \right| \qquad \varphi \in \left[ \frac{-\pi}{2} \quad \frac{\pi}{2} \right]$$

$tg(arctgx)=x$ и $arctg(tgy)=y$.

## 1.2 Метричен елемент.

### 1.2.1 Пресмятане дължина на отсечка.

Нека спрямо декартовата координатна система Oxy




$A_1(x_1,y_1)$ , $A_2(x_2,y_2)$.

Нека $d=|A_1A_2|$.

$$d^2=(x_2-x_1)^2+(y_2-y_1)^2$$

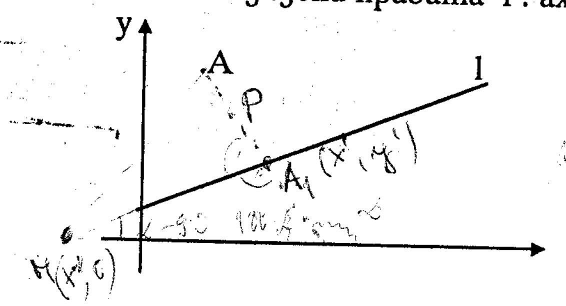
$$d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

Функция, написана на езика Pascal за пресмятане дължина на отсечка, ако са въведени координатите на краищата и :

```
function DD(x1,y1,x2,y2:real): real;
  begin
    DD=sqrt(sqr(x2-x1)+sqr(y2-y1));
  end;
```

### 1.2.2 Разстояние от точка до права.

Нека са дадени правата $l : ax+by+c=0$ и точката $A(x_1,y_1)$.



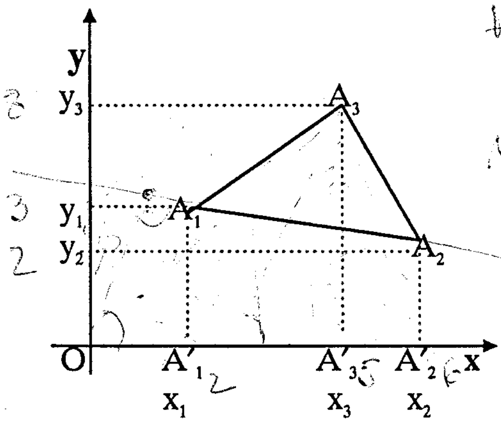

180 - (180 + α - 90 =

= α - 90

Разстоянието от т. A до правата l се пресмята по формулата :

$$\delta = \frac{|ax_1+by_1+c|}{\sqrt{a^2+b^2}}$$

### 1.2.3 Пресмятане лице на триъгълник по зададени координати на върховете му $A_1(x_1,y_1)$, $A_2(x_2,y_2)$, $A_3(x_3,y_3)$




Намиране норм. ур-ние през $A_1(2,3)$ и $A_2(6,2)$
Изхождаме от следн.
$Ax+By+C=0$
$A= y_2-y_1=2-3=-1$
$B=x_1-x_2=2-6=-4$
$-x-4y+C=0$

Нека триъгълникът $A_1A_2A_3$ е зададен с координатите на върховете си. Формулата за пресмятане лицето на триъгълника се извежда по следния начин [7]:

Разглеждаме няколко случая, взависимост от взаимното положение на точките $A_1$ , $A_2$ и $A_3$.

Нека $A_1A'_1 \perp Ox$ , $A_2A'_2 \perp Ox$ , $A_3A'_3 \perp Ox$

$S_{A1A2A3}=S_{A1A'1A'3A3}+ S_{A'3A'2A2A3} - S_{A1A'1A'2A2}$

формулу

$$\left.\begin{aligned} S_{A1A'1A'3A3} &= 1/2(x_3-x_1)(y_1+y_3) \\ S_{A'3A'2A2A3} &= 1/2(x_2-x_3)(y_2+y_3) \\ S_{A1A'1A'2A2} &= 1/2(x_2-x_1)(y_1+y_2) \end{aligned}\right| \Rightarrow$$

$S_{A1A2A3}=1/2 \, [ \, (x_3-x_1)(y_1+y_3)+(x_2-x_3)(y_2+y_3)-(x_2-x_1)(y_2+y_1) \, ]$
$S_{A1A2A3}=1/2 \, [ \, (x_3-x_1)(y_1+y_3)+(x_2-x_3)(y_2+y_3)+(x_1-x_2)(y_2+y_1) \, ]$

Аналогично разглеждане на останалите случаи показва, че формулата остава в сила винаги, когато точките $A_1A_2A_3$ взети в този ред определят положителна посока (обратна на движението на часовниковата стрелка) .

Когато $A_1$ ,$A_2$,$A_3$ определят отрицателна посока, тогава :

$S=-1/2 \, [ \, (x_3-x_1)(y_1+y_3)+(x_2-x_3)(y_2+y_3)+(x_1-x_2)(y_1+y_2) \, ]$

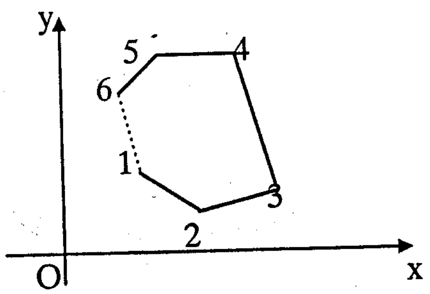
Полагаме $x_4=x_1$ и $y_4=y_1$ и обобщаваме:

$$S=1/2 \left| \sum_{i=1}^{3} (x_i-x_{i+1})(y_i+y_{i+1}) \right|$$

Функция, написана на езика Pascal, реализираща пресмятането :

```
Function SS(var X,Y:mas): real;
var i:byte;
     s:real;
   begin
    s=0.0;
    for I:=1 to 3 do
     s=s+(x[i]-x[i+1])*(y[i]+y[i+1]);
    SS:=1/2*ABS(s);
   end;
```

## 1.2.4 Пресмятане лицето на прост многоъгълник.




Нека е даден многоъгълникът $M=M_1M_2...M_n$. $M_i(x_i,y_i)$ , $i=1,2,..,n$
Полагаме $x_{n+1}=x_1$ , $y_{n+1}=y_1$.

По формулите за ориентираните лица на трапеците и тяхното сумиране получаваме формулата за лице на прост многоъгълник :

$$S=1/2 \left| \sum_{i=1}^{n} (x_i-x_{i+1})(y_i+y_{i+1}) \right|$$

```
Function SSM( N:word; var X,Y : mas):real;
var i:byte;
     s:real;
       begin
             s=0.0;
             for I:=1 to n do
             s:=s+(x[i]-x[i+1]) * (y[i]+y[i+1]);
             SSM:=1/2 *abs(s);
       end;
```

## 1.3 Комбинаторен елемент.

Елементарни комбинаторни обекти (съединения) наричаме такива наредени групи от какви да са предмети, наречени елементи, които се отличават една от друга или по реда на елементите или по самите елементи ([1], [8]). Например ако от десет различни елемента $A_1$, $A_2$,...,$A_{10}$ съставим групи от по няколко елемента като например $A_1 A_2 A_3$ , $A_2 A_1 A_3$ , $A_3 A_4 A_7 A_9$ , $A_1 A_2$ и т.н. се получават различни съединения на тези елементи. Някои от тях се различават само по реда на елементите $A_1A_3A_2$ и $A_2A_1A_3$, а други по влизащите в тях елементи, и дори по броя на елементите. Основните елементарни комбинаторни обекти са комбинации вариации и пермутации. Ще разглеждаме само съединения без повторение. За пораждане елементите на всяко едно от тези съединения се използва лексикографската наредба. За основно множество от което ще се пораждат комбинаторните съединения е удобно да се вземе множеството {1,2,...,n}, тъй като тези елементи са индекси на елементите $a_1,a_2,...,a_n$ и е достатъчно да получим тяхната наредба, за да получим наредбата на елементите $a_1,a_2,...,a_n$.

**Лексикографско подреждане** е такова подреждане, при което ако съединенията се разглеждат като числа , то те са подредени в нарастващ ред.

Ако $p=p_1p_2...p$ $a_1,a_2...a_n$ и $q=q_1q_2...q_n$ $p_i, q_i$ (1,2,...,n) , i=1,2,..., n то казваме, че p е лексикографично по малка от q, ако за някое k k=1,2,...,n-1 е изпълнено: $p_k=q_k$ и $p_{k+1}<q_{k+1}$.

Общата схема за получаване на лексикографската наредба може да се изрази така:

1.Пораждане на най-малкия елемент.

2.Пораждане на следващия по големина елемент и извеждането му.

3.Условие за прекратяване на пораждането.

### 1.3.1 Пермутации.

Пермутациите са комбинаторни съединения, в които участват всички елементи от множеството A={ $a_1,a_2...a_n$ } , n ≥ 1 ([4]).

Пермутациите се различават една от друга по местата на елементите $a_1,a_2,..,a_n$.

Всяко подреждане на елементите $a_1,a_2...a_n$ се нарича **пермутация.**

### Пермутации без повторение.

Пермутациите без повторение са пермутации, образувани от елементите на множеството A = { $a_1,a_2...a_n$ }, които са различни помежду си.

Нека означим броя на пермутациите с $P_i$. Достатъчно е да образуваме пермутациите на първите n естествени числа.

при n=1 1 ($P_1$=1)

при n=2 1 2 , 2 1 ($P_2$=2)

при n=3 всички възможни пермутации на числата 1,2 и 3 се получават по следния начин : към всяка от двете пермутации от два елемента се поставя числото 3 последователно отпред, по средата, и в края.

при n=3   3 1 2   1 3 2   1 2 3
          3 2 1   2 3 1   2 1 3

и т.н.

По същия начин от всяка пермутация от n-1 елемента, чрез поставянето на n на различни места се получават n пермутации.
Следователно $P_n=n.P_{n-1}$

$$\times \left|\begin{array}{l} P_1=1 \\ P_2=2P_1 \\ P_3=3P_2 \\ \dots\dots\dots\dots \\ P_{n-1}=(n-1)P_{n-2} \\ P_n= \quad nP_{n-1} \end{array}\right.$$

$$P_1.P_2.\ldots P_{n-1}.P_n=1.2.3.\ldots n.P_1.P_2.\ldots P_{n-1}$$

т.е. $P_n=n!$ е броят на пермутациите от n елемента.

Пермутацията, в която числата са подредени в естествен ред 1,2,3,...,n се нарича **основна.**

## Лексикографско пораждане на пермутации.

Алгоритъмът за лексикографско пораждане на пермутации без повторение от елементите на множеството {1,2,. . . ,n } се състои в следното :

1.Избираме начална пермутация **1, 2, 3, . . . n.**

2.Върху тази пермутация извършваме преобразувание и получаваме следващата в лексикографичен ред пермутация. Над новополучената пермутация отново извършваме същото преобразувание и получаваме нова пермутация докато се стигне до последната в лексикографичен ред пермутация **n, n-1, . . . 1.**

Ако $P=p_1\, p_2\, p_3 \ldots p_n$ е една пермутация, то следващата в лексикографичен ред получаваме чрез преобразуването:

1.Преглеждаме P отдясно наляво, за да намерим първата позиция i, такава, че $p_i<p_{i+1}$.

2.Търсим надясно от $p_i$ най-малкият елемент $p_j$ такъв, че $p_j>p_i$.

3.Сменяме местата на $p_i$ и $p_j$.

4.Извършваме транспозиция на елементите $p_{i+1}, p_{i+2} \ldots p_n$ т.е. сменяме местата им в порядък $p_n, p_{n-1}, \ldots, p_{i+1}$.

Пример:

Ako P=23876541 е една пермутация на елементите {1,2,...,8 }, то следващата в лексикографичен ред може да се намери така:

1. 23876541 i=2 , $p_2=3<8=p_3$
2. 23876541 j=7 , $p_7=4>3$ min { 8,7,6,5,4 }=4 сл. j=7
3. 24876531 сменяме местата на $p_i$ и $p_j$.
4. 24135678 транспозиция на елементите $p_3p_4 \ldots p_8$ в $p_8p_7 \ldots p_3$.

q=24135678 е следващата в лексикографичен ред пермутация.

## Програма за лексикографско пораждане на пермутации

```
program pp4;
type Masiv=array[1..100] of word;
var P:masiv;
   n,i:word;
     f:longint;
procedure Perm( N:word; Var P:Masiv);
Var M,K,J,L,K1 :word;
  Begin
  for k:=N-1 downto 1 do
    begin
     if p[K]<p[k+1] then
       begin
         m:=p[k+1];k1:=k+1;
          for j:=k+1 to n do
            begin
             if p[j]>p[k] then
               if p[j]<M then
                 begin
                  m:=p[j];k1:=j;
                       end;
             end;
               M:=p[k];p[k]:=p[k1];p[k1]:=M;
                     L:=N;K1:=K+1;
               repeat
                  M:=P[L];P[L]:=P[K1];P[K1]:=M;
                  L:=L-1;K1:=K1+1;
               until L<=K1;
           end;
       end;
    for i:=1 to n do
    write(p[i],  );
     writeln;
   end;
```

```
begin { main }
    read(n);
      f:=1;
      for i:=1 to n do
      f:=f*i;write(f);writeln;
        for i:= 1 to n do
        p[i]:=i;
       for i:=1 to n do
           write(p[i],  );writeln;
           { for i:= 1 to f do}
           perm(n,p);
end.
```

### 1.3.2. Вариации. Комбинации. Реалзация на N вложени цикъла.

Вариации.

**Вариациите** са комбинаторни съединения, в които участват k на брой елементи от множеството $A=\{a_1,a_2,...,a_n\}$ , $n \geq 1$.

Вариациите се различават една от друга по мястото на елементите си, или по състава си.

Вариациите без повторение са комбинаторни съединения от k различни елементи (k=1,2,...,n) от множеството $A=\{a_1,a_2,...,a_n\}$. Броят на вариациите

без повторение е $$V_n^k = \frac{n!}{(n-k)!}$$

Комбинации.

**Комбинациите** са комбинаторни съединения, в които участват k на брой елементи $k \geq 1$ от множеството $A=\{a_1,a_2,...,a_n\}$ , $n \geq 1$ ( [4] ).

Комбинациите се различават една от друга само по състава си.

**Комбинациите без повторение** са комбинаторни съединения от k различни елемента (k=1,2,. . .,n) от множеството $A=\{a_1,a_2,...,a_n\}$.

Комбинациите с повторение са комбинаторни съединения от k елементи (при $k \geq 1$). Някои елементи могат да участват и повече от един път (от множеството $A=\{a_1,a_2,...,a_n\}$).

Различието в местата на елементите не обуславя различни комбинации (напр. $a_1a_2$ и $a_2a_1$ са една и съща комбинация).

Когато трябва да изберем k елемента от n , казваме, че комбинацията е комбинация на n елемента от клас k.

Например елементите $a_1a_2a_3a_4$ могат да образуват само 6 комбинации от клас 2 :

$$\begin{array}{lll} a_1a_2 & a_1a_3 & a_1a_4 \\ & a_2a_3 & a_2a_4 \\ & & a_3a_4 \end{array}$$

От същите елементи се получават следните комбинации от 3 елемента : $a_1a_2a_3$ $a_1a_2a_4$ $a_1a_3a_4$ $a_2a_3a_4$.

Броят на комбинациите от n елемента от клас k Cnk получаваме по следния начин : като пермутираме елементите от всяка комбинация ще получим по толкова съединения, колкото е броят на вариациите от k елемента.

$$C_n^k \cdot P_k = V_n^k$$

$$C_n^k = \frac{V_n^k}{P_k} = \frac{n(n-1)\ldots(n-k+1)}{1.2\ldots k} \cdot \frac{(n-k)!}{(n-k)!}$$

$$C_n^k = \frac{n!}{k!\,(n-k)!}$$

От всички n елементи може да се образува само една комбинация, съдържаща всички елементи.

Комбинациите притежават свойството $C_n^k = C_n^{n-k}$.

Това равенство може да се получи като вземем предвид, че на всяка комбинация от клас k отговаря една комбинация от клас n-k (тази, която е образувана от останалите елементи) и обратно. Това свойство се използва за намиране броя на комбинациите, когато класът им е число, по-голямо от половината от броя на елементите. Например $C_{20}^{18} = C_{20}^{2} = 190$

$$C_n^k = C_{n-1}^k + C_{n-1}^{k-1} \qquad C_n^k = \binom{n}{k}$$

**Избор на два елемента от n .**

i,j  i<j  $1 \le I, j \le n$  =>  $1 \le i \le n-1$

$I+1 \le j \le n$

```
For  I:= 1 to n-1  do
  For  j:=i+1  to   n   do
  .....................................
```

**Избора на три елемента от n** може да се реализира с три вложени цикъла:

$1 \le i < j < k \le n$

$1 \le i \le n-2$

$I+1 \le j \le n-1$

$j+1 \le k \le n$

```
For I:=1  to  n-2  do
  For  j:=i+1  to  n-1   do
    For k:=j+1   to   n   do
    . . . . . . . . . . . . . . .
```

Лексикографското пораждане на комбинации без повторение започва с началната комбинация 1, 2,..., k и завършва с комбинацията n-k+1,..., n.

Преобразуването, с помощта на което се получава всяка следваща комбинация се състои от следните елементарни действия:

1.Преглеждане на текущата комбинация отдясно наляво и определяне мястото на най-десния елемент, който не е достигнал максималната си стойност.

2.Увеличаване на този елемент с 1, а на всички елементи надясно от него се дават нови най-малки възможни стойности (елементите в комбинациите се разглеждат в нарастващ ред).

**Схема за получаване на комбинации без повторение** от три елемента от множеството 1 2 3 4 5 6, като се използува алгоритъма за лексикографско пораждане:

123 → 134 → 145 → 156 → 234 → 245 → 345 → 356 → 456

↓ 124, ↓ 125, ↓ 126

↓ 135, ↓ 136

↓ 146

↓ 235, ↓ 236

↓ 246

↓ 346

n-2 n-1 n

n=6 k=3

като със стрелка се посочва следващата комбинация , получена по правилото 1, а с точка - увеличения с 1 елемент по правилото 2.

Пример:

При избор на четири елемента от елементите 1 2 3 4 5 6 се получават следните комбинации: k=4

| i1 | i2 | i3 | i4 |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 1 | 2 | 3 | 5 |
| 1 | 2 | 3 | 6 |
| 1 | 2 | 4 | 5 |
| 1 | 2 | 4 | 6 |
| 1 | 2 | 5 | 6 |
| 1 | 3 | 4 | 5 |
| 1 | 3 | 4 | 6 |
| 1 | 3 | 5 | 6 |
| 1 | 4 | 5 | 6 |
| 2 | 3 | 4 | 5 |
| 2 | 3 | 4 | 6 |
| 2 | 3 | 5 | 6 |
| 2 | 4 | 5 | 6 |
| 3 | 4 | 5 | 6 |

k=4 n=6

n-k+1

## Алгоритъм за лексикографско пораждане на комбинации.

Генериране на всички k елементни подмножества на n елементното множество X={1,2,. . . ,n}.

На всяко k елементно подмножество взаимно-еднозначно съответствува нарастваща последователност с дължина k елемента от X.

Например на подмножеството {3,5,1} съответствува последователността {1,3,5}.

Следваща за комбинацията $< a_1a_2 \ldots a_k >$ се явява $< b_1b_2 \ldots b_k > = < a_1 \ldots a_{p-1}\, a_p+1\, a_p+2 \ldots a_p+k-p+1 >$, където $p=\max\{ i : a_i<n-k+1 \}$ , а следващата последователност след $< b_1b_2 \ldots b_k >$ е $< b_1 \ldots b_{p-1}\, b_p+1\, b_p+2 \ldots b_p+k-p+1 >$ където :

$$p = \begin{cases} p-1 & \text{ako } b_k=n \\ k & \text{ako } b_k<n \end{cases},$$

като $< a_1a_2 \ldots a_k >$ и $< b_1b_2 \ldots b_k >$ са различни от $< n-k+1 \ldots n >$ - последната в лексикографичен ред пермутация.

**Програма, реализираща лексикографско пораждане на комбинации:**

```
Program PP10;   { Лексикографско пораждане на комбинации }
uses Crt;
Type Masiv=array[1..50] of integer;
Var  X :Masiv;
   N,K :word;

procedure KOMB(N,K:word;VAR A:Masiv);  { генерира всички комбинации }
  Var i,PP,L:word;
   begin
    for i:=1 to K do
     A[i]:=i;
           PP:=K;
        while PP>=1 do
         begin
         for L:=1 to K do
          write(A[L],' ');  { Намерена е поредната комбинация }
          writeln;
         If A[K]=N then PP:=PP-1
                   else PP:=K;
           if PP>=1 then
            for i:=K downto PP do
             A[i]:=A[PP]+i-PP+1;
         end;
    end;
    begin { main }
    ClrScr;
     write('N=');read(N);
     write('K=');read(K);
     KOMB(N,K,X);
    end.
```

## Реализация на N вложени цикъла .

Нека е дадено естествено число N и числа $a_i$ , $b_i$ $a_i \le b_i$ $i=1,2,\ldots,n$.
Да се вложат N цикъла един в друг,като в най-вътрешния се отпечатат стойностите на параметрите на циклите.

$$a_1 \le i_1 \le b_1$$
$$a_2 \le i_2 \le b_2$$
$$\vdots$$
$$a_n \le i_n \le b_n$$

Описание на алгоритъма:

1.На параметрите на циклите присвояваме началните им стойности ($i_j:=a_j$ ), където $j=1,2,\ldots,n$.

2.На j присвояваме начална стойност n и проверяваме, дали $i_j$ е достигнало крайната си стойност $b_j$. Ако това условие е изпълнено (т.е. най-вътрешния цикъл е завършил), проверяваме дали параметъра на следващия по-външен цикъл е приел всичките си стойности до $b_j$ .

Ако условието не е изпълнено, тогава увеличаваме $i_j$ с 1 и проверяваме дали j<n.

Ако j<n на всички параметри на по-вътрешните цикли (от $i_{j+1},\ldots,i_n$) присвояваме началните им стойности и отново проверяваме, дали най-вътрешния цикъл е завършил.Тези действия се повтарят докато j стане по-малко от 1.

## Програма, реализираща N вложени цикъла

```
Program Cikli; {Реализация на N вложени цикъла
uses crt;
type mas=array[1..50] of integer;
var Br,i,j,n,k,l:integer;
        a,b,ii:mas;
            c:Char;
  begin
  ClrScr;
 write( n= );  readln(n);
 for i:=1 to n do
  begin
    write( a ,i, = );readln(a[i]);ii[i]:=a[i]
    write( b ,i, = );readln(b[i]);
  end;
  j:=n;  ClrScr;Br:=2;
  REPEAT
  While ii[j]<b[j] DO
```

```
    begin
for k:=1 to n do write(ii[k]);
      writeln;Br:=Br+1;if Br>=24 then  begin
                      c:=Readkey;br:=2;Clrscr;
                      end;
      ii[j]:=ii[j]+1;
      if j<n then for l:=j+1 to n do ii[l]:=a[l];
      j:=n;
    end;
    j:=j-1;
  UNTIL j<1;
  for k:=1 to n do write(ii[k]);
  writeln;
  repeat until keypressed;
end.
```

## 1.4. Приложение.

### Често използвани подпрограми

#### ПП за въвеждане на цяло число

```
 Procedure IIK(d,g:word;Var RESS:word)
   Var  s :string[80];
   Kod,X,Y:word;
  begin
   write( Задайте цяло положително число от интервала [ ,d, , ,g, ]= );
   X:=WhereX;Y:=WhereY;
    repeat
     write(^G);GotoXY(X,Y);ClrEol;
      readln(s);
       Val(s,Ress,Kod);
       if Kod<>0 then write(^G);
      until (Kod=0) AND (RESS IN [d..g]);
  end;
```

**ПП за въвеждане координатите на n точки в два масива**

```
Procedure KOORD(Var m:integer;Var XX,YY:Masiv);
Var i:integer;
  Procedure InputIntegerKod(Var Res:integer);
  Var     s:string[80];
    Kod,X,Y:integer;
     begin
      write( Въведете цяло число: );
      X:=WhereX;Y:=WhereY;
        repeat
         write(^G);GotoXY(X,Y);ClrEol;
         Readln(s);
         Val(S,Res,Kod);
        until Kod=0;
    end;
     Begin
       write( Въведете броя на точките: );
       repeat
       InputIntegerKod(m);
 until m>1;
      writeln( Въведете координатите на точките: );
      for i:=1 to m do
        begin
         write( X( ,i, )= );read(XX[i]);
         write( Y( ,i, )= );read(YY[i]);
        end;
    end;
```

Функция за пресмятане дължина на отсечка

```
function dd(x1,y1,x2,y2:real):real;
  begin
    dd:=sqrt(sqr(x2-x1)+sqr(y2-y1));
   end;
```

Функция за намиране лице на триъгълник по зададени координати на върховете му

```
function SS(var x,y:mas):real;
   var i:byte;
          s:real;
       begin
       S:=0.0;
       for i:=1 to 3 do
         S:=S+(x[i]-x[i+1])*(y[i]+y[i+1]);
           SS:=1/2*ABS(S);
       end;
```

**Функция за намиране лице на многоъгълник по зададени координати на върховете му**

```
function SSM(N:word; Var x,y:mas):real;
var i:word;
      S:real;
  begin
  S:=0.0;
  for i:=1 to N do
    S:=S+(x[i]-x[i+1])*(y[i]+y[i+1]);
     SSM:=1/2*ABS(S);
  end;
```

Процедури за намиране на минимален и максимален елемент на масив

```
procedure Max(n:word;var a:masiv;var am:real);
   var l:integer;
     Begin
      am:=a[1];    for l:=2 to n do
                    if a[l]>am then am:=a[l];
     end;
procedure Min(n:word;var a:masiv;var an:real);
   var l:integer;
     Begin
      an:=a[1];
      for l:=2 to n do
```

```
    if a[l]<an then an:=a[l];
  end;
```

Функция ,която проверява дали един многоъгълник е изпъкнал

```
Function IZP(nn:word;Var u,v:masiv):boolean;
  var k,i,j,bp,bo:word;

    function  SGN(xx,yy:real):integer;
     var f:real;
      begin
       f:=(xx-u[i])*(v[j]-v[i])-(yy-v[i])*(u[j]-u[i]);
       if f=0 then sgn:=0;
       if f<0 then sgn:=-1
         else sgn:=1;
      end;{SGN}

    Begin   {IZP}
      for i:=1 to nn do
        begin
        j:=i+1;bp:=0;bo:=0;
          for k:=1 to nn do
           begin
                       if j=nn+1 then j:=1;
        if (k<>i) and (k<>j) then
            CASE SGN(u[k],v[k]) OF
                1:bp:=bp+1;
               -1:bo:=bo+1;
            end;
         end;
       if (bp=nn-2) or (bo=nn-2) then IZP:=true
          else begin
                IZP:=false;
                exit;
              end;
       end;  {i}
    end;{IZP}
```

## Процедура за лексикографско пораждане на пермутации

```
  Procedure Perm(n:word;Var P:Masiv);
 Var i,j,min,imin,m,l :word;
   begin
        i:=n-1;
  while NOT (p[i]<p[i+1]) do   i:=i-1;
  min:=p[i+1];imin:=i+1;
  for j:=i+1 to n do
    begin
     if p[j]>p[i] then
      if p[j]<min then
        begin
          min:=p[j];imin:=j;
        end;
    end;
  m:=p[i];p[i]:=min;p[imin]:=m;
  j:=i+1;l:=n;
  repeat
   m:=p[j];p[j]:=p[l];p[l]:=m;
   j:=j+1;l:=l-1;
  until j>=1;
 end;
```

## Програма за лексикографско пораждане на комбинации

```
Program PP10;   { Лексикографско пораждане на комбинации }
uses Crt;
Type Masiv=array[1..50] of integer;
Var  X :Masiv;
  N,K :word;

procedure KOMB(N,K:word;VAR A:Masiv);  { генерира всички комбинации }
 Var i,PP,L:word;
  begin
   for i:=1 to K do
    A[i]:=i;
    PP:=K;
     while PP>=1 do
      begin
       for L:=1 to K do
        write(A[L],   ); { Намерена е поредната комбинация }
        writeln;
        If A[K]=N then PP:=PP-1
                        else PP:=K;
          if PP>=1 then
           for i:=K downto PP do
```

```
            A[i]:=A[PP]+i-PP+1;
       end;
  end;
    begin { main }
ClrScr;
     write( N= );read(N);
     write( K= );read(K);
     KOMB(N,K,X);
    end.
```

Програма, реализираща N вложени цикъла

```
Program Cikli; {Реализация на N вложени цикъла}
Uses Crt;
type mas=array[1..50] of integer;
var Br,i,j,n,k,l:integer;
         a,b,ii:mas;
             c:Char;
   begin  ClrScr;
  write( n= );   readln(n);
  for i:=1 to n do
   begin
     write( a ,i, = );readln(a[i]);ii[i]:=a[i];
     write( b ,i, = );readln(b[i]);
   end;
   j:=n;  ClrScr;Br:=2;
   Repeat
   While ii[j]<b[j] Do
     begin
             for k:=1 to n do write(ii[k]);
              writeln;Br:=Br+1;if Br>=24 then
                     begin
                       c:=Readkey;br:=2;Clrscr;
                     end;
          ii[j]:=ii[j]+1;
          if j<n then for l:=j+1 to n do ii[l]:=a[l];
          j:=n;
       end;
     j:=j-1;
    Until  j<1;
    for k:=1 to n do write(ii[k]);
    writeln;
    repeat until keypressed;
 end.
```

```
UNIT  First;

INTERFACE
Uses Crt;
Type masiv=array[1..50] of real;
     mas=array[1..50] of word;
{Var n,z:word;
   p:mas;}
 Function IZP(nn:word;Var u,v:masiv):boolean;
 Procedure KOORD(Var m:word;Var X1,Y1:masiv);
 Procedure Perm(n:word;Var p:mas);
 Function Fac(z:word):word;
 Procedure IIK(d,g:word;Var RESS:word);
 Function DD(XX1,YY1,XX2,YY2:real):real;
 Function SSM(N:word;Var X,Y:Masiv):real;

 IMPLEMENTATION

     Procedure IIK;
 Var  s :string[80];
        Kod,X,Y:word;
       begin
        write( Задайте цяло положително число от интервала [ ,d, , ,g, ]= );
        X:=WhereX;Y:=WhereY;
         repeat
          write(^G);GotoXY(X,Y);ClrEol;
           readln(s);
            Val(s,Ress,Kod);
            if Kod<>0 then write(^G);
           until (Kod=0) AND (RESS IN [d..g]);
        end;

      Procedure KOORD;
  Var i:word;
      Procedure InputIntegerKod(Var Res:word);
      Var  s :string[80];
      Kod,X,Y:word;
  begin
      X:=WhereX;Y:=WhereY;
      repeat
        write(^G);GotoXY(X,Y);ClrEol;
        readln(s);
        Val(s,Res,Kod);
      until Kod=0;
     end;
        begin
          write( Въведете броя на точките : );
            repeat
```

```
            InputIntegerKod(m);
              until m>2;
              writeln( Въведете координатите на точките : );
              for i:=1 to m do
                begin
                 write ( X( ,i, )= );read(X1[i]);
                 write ( Y( ,i, )= );read(Y1[i]);
                end;
            X1[m+1]:=X1[1];Y1[m+1]:=Y1[1];
end;

function DD;
  begin
   dd:=sqrt(sqr(xx2-xx1)+sqr(yy2-yy1));
  end;

 function SSM;
   var i:word;
       S:real;
     begin
      S:=0.0;
      for i:=1 to N do
       S:=S+(x[i]-x[i+1])*(y[i]+y[i+1]);
      SSM:=1/2*ABS(S);
     end;

Function IZP;
   var k,i,j,bp,bo:word;

     function  SGN(xx,yy:real):integer;
      var f:real;
       begin
        f:=(xx-u[i])*(v[j]-v[i])-(yy-v[i])*(u[j]-u[i]);
        if f=0 then sgn:=0;
        if f<0 then sgn:=-1
          else sgn:=1;
      end;{SGN}

     Begin   {IZP}
      for i:=1 to nn do
        begin
        j:=i+1;bp:=0;bo:=0;
          for k:=1 to nn do
           begin
                       if j=nn+1 then j:=1;
         if (k<>i) and (k<>j) then
            CASE SGN(u[k],v[k]) OF
```

```
              1:bp:=bp+1;
             -1:bo:=bo+1;
           end;
         end;
        if (bp=nn-2) or (bo=nn-2) then IZP:=true
          else begin
              IZP:=false;
              exit;
             end;
        end;  {i}
      end;{IZP}

     Procedure Perm;
    Var i,j,min,imin,m,l :word;
       begin
           i:=n-1;
    while NOT (p[i]<p[i+1]) do   i:=i-1;
    min:=p[i+1];imin:=i+1;
    for j:=i+1 to n do
       begin
        if p[j]>p[i] then
         if p[j]<min then
           begin
             min:=p[j];imin:=j;
           end;
       end;
    m:=p[i];p[i]:=min;p[imin]:=m;
    j:=i+1;l:=n;
    repeat
     m:=p[j];p[j]:=p[l];p[l]:=m;
     j:=j+1;l:=l-1;
    until j>=1;
   end;

  Function Fac;
     Var i,f:word;
       begin
        f:=1;
        for i:=1 to z do
        f:=f*i;
        Fac:=f;
       end;
end.     { First }
```

## 2. Задачи.

### 2.1 Задачи от изпъкналост на геометрични фигури.

Един многоъгълник е **изпъкнал**, ако за всеки два съседни върха е вярно, че всички останали върхове лежат в една и съща полуравнина, относно правата, определена от двата върха.

Нека М е многоъгълник, зададен с координатите на на върховете си $A_1, A_2, \ldots, A_n$. Проверката дали многоъгълникът е изпъкнал може да се извърши по следния алгоритъм :

1.За удобство означаваме точката $A_{n+1}=A_1$.

2.За всеки връх $A_i$, $i=1,\ldots,n$ се изпълнява следното:

-определя се общото уравнение на правата $A_iA_{i+1}$;

-проверява се дали останалите върхове различни от точките $A_i$ и $A_{i+1}$ лежат в една и съща полуравнина относно правата $A_iA_{i+1}$.

Този алгоритъм се прилага при решаването на задача 2.

**1.Задача.** В равнината са дадени N точки, зададени с координатите на върховете си. Да се определи реда, в който можем да съединим точките така, че да получим прост n-ъгълник (без самопресичане).

**2 Задача.** Дадени са координатите на върховете на прост многоъгълник $M=M_1M_2\ldots M_n$. Да се провери дали многоъгълникът е изпъкнал.

**Решение:**

<u>Описание на алгоритъма:</u>

1.Въвеждат се броя и координатите на точките.

2.Проверява се дали многоъгълникът М е изпъкнал. Върховете му са взети в този ред.

Проверката се извършва с помощта на функцията IZP, описана в модула First.

3.Отпечатва се подходящо съобщение.

```
Program I1;
Uses Crt,First;
 Var A,B :masiv;
        N:word;
     begin
         KOORD(N,A,B);
         if IZP(N,A,B)= true then
               begin
                  write( Многоъгълникът е изпъкнал );exit;
               end
                 else
                  begin
                   write( Многоъгълникът не е изпъкнал. );exit;
                  end;
     end.
```

**3. Задача** В равнината са дадени точките $M_1, M_2, ..., M_n$, зададени с координатите на върховете си $M_i(x_i, y_i)$. Да се провери дали съществува изпъкнал многоъгълник с върхове в тези точки, и ако съществува, да се определи обхода на върховете.

**Решение:**

Описание на алгоритъма:

1.Въвеждат се броя и координатите на точките.

2.За намиране на всички n-ъгълници с върхове точките $M_1, M_2, ..., M_n$ е необходимо да се намерят всички обходи на тези точки, т.е. всички пермутации на тези елементи.

3.За всеки n-ъгълник се проверява дали е изпъкнал.

4.Програмата отпечатва всички възможни изпъкнали n-ъгълници с върхове дадените n точки.

```
Program I2;
Uses Crt,First;
var p1:mas;
   i1,n1,j1,k1,Br,o :word;
       t,s,xc,yc:masiv;
  begin { main }
  ClrScr;Br:=0;
   KOORD(n1,xc,yc); {във. на координати}
   for i1:=1 to n1 do p1[i1]:=i1;
     t:=xc;s:=yc; {присвояване на цели масиви}
       if IZP(n1,t,s) = true then
          begin
           write( Едно решение са точките с координати :  );
            for i1:=1 to n1 do
            write( ( ,t[i1]:5:2, , ,s[i1]:5:2, )    );
            Br:=Br+1;
               end;
      for i1:=1 to fac(n1)-1 do
        begin
         perm(n1,p1);for o:=1 to n1 do write(p1[o]);write(
          for j1:=1 to n1 do
            begin
             t[j1]:=xc[p1[j1]];
             s[j1]:=yc[p1[j1]];
            end;
             if IZP(n1,t,s)= true then
     begin
                 Br:=Br+1;
                 for o:=1 to n1 do write(p1[o]);
                 write(Br, -то решение : );
                  for k1:=1 to n1 do
                  write( ( ,t[k1]:5:2, , ,s[k1]:5:2, ) ,     );
      writeln;
```

```
            end;
                      end;{i1}
                writeln( Броят е -  ,Br);writeln(fac(n1));write(n1);
      end.
```

**4 Задача.** Да се построи множеството на всички различни изпъкнали p-ъгълници с върхове в зададено множество от n точки.

**Решение:**

Описание на алгоритъма:

1.Въвеждат се броя и координатите на точките.

2.Генерират се всички възможни p-ъгълници (комбинации на p елемента от n), чиито върхове са измежду дадените n точки.

3.За всеки p-ъгълник се проверява дали е изпъкнал (с функцията IZP) .

4.Отпечатват се координатите на върховете на всеки p-ъгълник, който е изпъкнал.

```
program I3;
Uses Crt,First;
Var br,pp,p,i,n :word;
      t,s,x,y:masiv;
         a:mas;
  begin  { main }
    KOORD(n,x,y);br:=0;
      repeat
      write( p= ); read(p); {проверка за цяло тр. }
      until p>2;
      for i:=1 to p do   a[i]:=i;
        pp:=p; { komb }
        while pp>=1do
          begin
           for i:=1 to p do
             write(a[i],   );
                for i:=1 to p do
          begin
            t[i]:=x[a[i]];s[i]:=y[a[i]];
           end;
           t[p+1]:=t[1];s[p+1]:=s[1];
           if IZP(p,t,s)=true then
             begin  br:=br+1;
              writeln( Намерен изпъкнал ,p, -ъгълник с върхове  : )
              for i:=1 to p do write( ( ,t[i]:3:1, , ,s[i]:3:1, )
             end;
           if a[p]=n then pp:=pp-1
                  else pp:=p;
           if pp>=1 then for i:=p downto pp do
         a[i]:=a[pp]+i-pp+1;
```

```
        end;
        write( br= ,br);
    end.
```

**5 Задача.** Дадени са n положителни числа $\{a_i\}$ i=1,2,. . . ,n. Съществува ли изпъкнал n-ъгълник с дължини на страните си $\{a_i\}$ , i=1,2,. . .,n. Променя ли се решението ако става дума за прост многоъгълник?

**6 Задача. / Конкурсна задача сп."Математика" бр.9 1985 г. /**

Нека $M_1(x_1,y_1)$ ,$M_2(x_2,y_2)$ , ..., $M_n(x_n,y_n)$ $4 \leq n \leq 20$ са n точки в равнината, чиито координати $x_1,y_1,x_2y_2,. . . ,x_n,y_n$ са зададени като последователни елементи на едномерен масив А. Известно е, че при подходящо подреждане тези точки са последователни върхове на изпъкнал многоъгълник.

Да се състави програма,която:

а) да въвежда стойността на променливата n и координатите на n-те точки;

б) да пренарежда елементите на масива А във вида $(x_{i1},y_{i1},x_{i2},y_{i2},...,,x_{in},y_{in})$, така, че $M_{i1},M_{i2},. . .,M_{in}$ да е изпъкнал многоъгълник, където $x_{ik},y_{ik}$ са координати на точката $M_{ik}$ ;

в) да пресмята лицето на изпъкналия многоъгълник $M_{i1},M_{i2},. . .,M_{inм}$ определен в подточка б).

**Решение:**

<u>Описание на алгоритъма:</u>

За решение на задачата е използвана статията [7].

1.Въвеждат се стойността на n и кординатите на n-те точки, като се използува подпрограмата KOORD. Елементите на двата масива се прехвърлят в масив А.

2.С помощта на подпрограмата за намиране на пермутации Perm се намират всички обходи на върховете.

3.В масива ZZ се записва поредната пермутация на номерата на точките, които трябва да се вземат за върхове на многоъгълник в този ред.

4.За всеки такъв многоъгълник се проверява дали е изпъкнал.

5.Отпечатват се координатите на всички намерени изпъкнали многоъгълници.

Програмата отпечатва всевъзможните подреждания на точките, които са върхове на изпъкнали многоъгълници.

```
Program I5;
uses crt,First;
var  A,X,Y:masiv;
       ZZ:mas;
br,l,j,i,n:word;

      BEGIN {main}
      ClrScr;br:=0;
        KOORD(n,X,Y);
```

```
i:=1;
 repeat
  A[2*i-1]:=X[i];A[2*i]:=Y[i];
  i:=i+1;
 until i>n;
      for i:=1 to n do ZZ[i]:=i;
       if IZP(n,X,Y)=true then
         begin
          br:=1;
           write( Поредният изпъкнал мног.е с върхове с коорд.: );
               i:=1;
                 repeat
                    write( ( ,A[i]:2:0, , ,A[i+1]:2:0, )  );
                    i:=i+2;
                 until i>2*N;
                writeln;
          end;
          for l:=1 to fac(n)-1 do
              begin
               Perm(n,ZZ);
                { зареждане на мас X и У }
                for i:=1 to n do
                  begin
                    j:=ZZ[i];
                    X[i]:=A[2*j-1];
                    Y[i]:=A[2*j];
                  end;
                          if IZP(N,X,Y) then
                     begin
                     write( Поредният изпъкнал мног.е с върхове с коорд.: );
                               br:=br+1;
                                 for i:=1 to n do write( ( ,x[i]:2:0, , ,y[i]:2:0,
                                 writeln;
              end;
          end;
   write( Броят на изпъкналите многоъгълници е  ,br);
 end.
```

## 2.2 Метрични задачи.

### 2.2.1. Задачи, свързани с намиране дължина на отсечка и ъгъл между две прави.

**1 Задача.** Дадено е множество от n точки, зададени с координатите на върховете си. Да се намери дължината на най-късата (най-дългата) отсечка с краища някои от тези точки.

**Решение:**
<u>Описание на алгоритъма:</u>

1.Въвеждат се броя и координатите на точките.
2.От n-те точки се избират всевъзможните двойки точки (комбинации на 2 елемента от n). Този избор се реализира с 2 вложени цикъла.
3.Намират се разстоянията между тези двойки точки.
4.Извежда се мимималното от тези разстояния.

```
program M1;
Uses Crt,First;
Var         X,Y:Masiv;
  ii,ij,i,j,N  :word;
       D,Min  :Real;

    BEGIN  { main }
      Min:=1.7E38;
      Koord(N,X,Y);
       for I:=1 to N-1 Do
                   for J:=I+1 to N do
          begin
           d:=DD(X[i],Y[i],X[J],Y[j]);
            if Min>D then
                begin
                   Min:=D;
                    ii:=i;ij:=j;
                 end;
             end;
          write( Дължината на най-късата отсечка е ,Min:5:2, с краища
точките с );
          wrire( координати ( ,X[ii]:5:2, ,Y[ii]:5:2, ) и ( ,X[ij]:5:2, , ,Y[ij]:5:2, ). );
     END.
```

**2 Задача** От дадено множество от n точки да се намери такава точка, че сумата от разстоянията от нея до останалите да е минимална.

**Решение:**

Описание на алгоритъма:

1.Въвеждат се броя и координатите на точките.

2.За всяка точка се намират разстоянията от нея до всяка от останалите точки.

3.Тези разстояния се сумират.

4.Намира се минималната от тези суми.

```
Program M2; {Ако има няколко точки с търсеното св-во дава първата
намерена}
 Uses Crt,First;
  Var  x,y:masiv;
    im,j,i,n:word;
      min,s:real;
        BEGIN {main}
        ClrScr;
          min:=1.7E38;
            KOORD(n,x,y);
                      for i:=1 to n do
               begin
                 s:=0.0;
                 for j:=1 to n do
                  begin
                  if i<>j then s:=s+DD(x[i],y[i],x[j],y[j])
                  end;
                  if s<min then
                   begin
                    min:=s;im:=i;
                          end;
               end;
            writeln('Търсената точка е с номер ',im);
            writeln('Координатите и са (',x[im]:5:2,',',y[im]:5:2,')');
            writeln('Търсеното минимално разстояние е ',min:5:2)
 end.
```

**3 Задача** Нека $P_1(x_1,y_1),P_2(x_2,y_2),\ldots,P_n(x_n,y_n)$ са n точки от равнината. Да се състави процедура, която подрежда точките по такъв начин, че разстоянията им до началото на координатната система да образуват ненамаляваща редица.

**4 Задача/ Първа републиканска олимпиада - Украйна май 1988 г.-** сп."Математика" бр.10 1988 г. /

На една права са оцветили **n** отсечки. Известни са координатите L[I] на левия край и координатите R[I] на десния край на i-тата отсечка за i=1,2,..,n. Намерете сумата от дължините на всички оцветени части на правата.

Забележка: Числото n е толкова голямо, че при изпълнение на даже $n^2$ прости операции не достига машинно време.

**5 Задача.** Разстоянието между две множества от точки в равнината - това е разстоянието между най-близко разположените точки от тези множества. Да се намери разстоянието между две зададени множества от точки.

**6 Задача Конкурс на младите програмисти на литовската ССР - 1985 г.** сп."Математика" бр.5 1988 г.

Дадени са дължините на страните на правоъгълник, изразени в цели числа.От единия връх на правоъгълника се прекарва линия, сключваща със страните на правоъгълника ъгли по 45 градуса. Достигайки една от страните, линията се пречупва под ъгъл 90 градуса. Достигайки друга страна, тя отново се пречупва под ъгъл 90 градуса и т.н. докато не достигне някой от върховете на правоъгълника. Получената начупена линия наподобява траекторията на билярдна топка по правоъгълна дъска.

Да се състави програма за определяне броя на точките на пречупване и дължината на начупената линия.

### 2.2.2. Задачи за намиране лица и периметри на многоъгълници.

**7 Задача** Съставете програма,която по зададени страни на триъгълник пресмята лицето му и радиуса на вписаната и описаната окръжност.

**8 Задача** Да се построи многоъгълник (не непременно изпъкнал) с върхове точките от зададено множество от точки в равнината и с максимален периметър.

**9 Задача** Да се намери лицето на прост многоъгълник, зададен с координатите на върховете си.

За решаването на тази задача, след въвеждане броя и координатите на точките, непосредствено се прилага подпрограмата за намиране лице на прост многоъгълник SSM .

```
Program M3;
Uses Crt,First;
Var X,Y:masiv;
    n:word;
     Begin { main }
       KOORD(n,X,Y);
       write( Лицето на многоъгълника е:  ,SSM(n,X,Y):5:2);
     end.
```

**10 Задача Конкурсна - сп."Математика" бр.4 1986 г**

Нека A и B са изпъкнали многоъгълници в една равнина, съответно с върхове $A_1,A_2,...,Am$ и $B_1,B_2,...,B_n$ , чиито координати са цели неотрицателни числа ( $n \geq 5$ , $m \geq 5$). Разположението на мнигоъгълниците A и B е такова, че върховете $A_2,A_3,...,A_k$ са вътрешни точки за многоъгълника B, а върховете $B_2,B_3,...,B_l$ са вътрешни точки за A $(1<k<m , 1<l<n)$, kато $A_1{\scriptstyle\blacksquare}B_{l+1}$ , $B_1{\scriptstyle\blacksquare}A_{k+1}$. Върховете и на двата многоъгълника

$A_1, A_2, \ldots, A_m$ и $B_1, B_2, \ldots, B_n$ са наредени в посока, обратна на движението на часовниковата стрелка. Да се състави програма,чрез която:

а) се въвеждат координатите на върховете на многоъгълниците A и B, след която многоъгълниците се изобразяват на екрана;

б) се пресмятат лицата на многоъгълниците A и B и на тяхното обединение, т.е. на многоъгълника с върхове $B_{l+1}, B_{l+2}, \ldots, A_{k+1}, A_{k+2}, \ldots, A_m$.

**Решение:**

Описание на алгоритъма:

За да се намери лицето на обединението на двата многоъгълника трябва:

1.Да се въведат стойности за M и N.

2.Да се въведат координатите на върховете на многоъгълниците A и B.

3.Да се въведат стойности за k и l.

4.Да се пресметнат лицата на двата многоъгълника A и B (подпрограмата SSM от модула First).

5.За пресмятане лицето на обединението се сумират лицата на многоъгълниците A и B и от тази сума се вади лицето на многоъгълника с върхове $B_1B_2 \ldots B_lB_{l+1}\ A_2A_3 \ldots A_kA_{k+1}$.

```
Program M7;
Uses Crt,First;
Var AX,AY,BX,BY,TX,TY  :Masiv;
       i,m,n,l,k :word;
             s :real;
    BEGIN  { main }
     write( Въведете броя на върховете на многоъгълниците A и B >=5  );
     Koord(m,AX,AY);AX[m+1]:=AX[1];AY[m+1]:=AY[1];
     Koord(n,BX,BY);BX[n+1]:=BX[1];BY[n+1]:=BY[1];
     write( Въведете k:[1,m] );IIK(1,m,k);
     write( Въведете l:[1,n] );IIK(1,n,l);
     writeln( Лицето на многоъгълника A е  ,SSM(m,AX,AY):5:2);
     writeln( Лицето на многоъгълника B е  ,SSM(n,BX,BY):5:2);
     S:=SSM(m,AX,AY)+SSM(n,BX,BY);
     for i:= 1 to l do
      begin
       TX[i]:=BX[i];TY[i]:=BY[i];
      end;
           or i:=l+1 to k+l+1 do
              begin
               TX[i]:=AX[i-l];
               TY[i]:=AY[i-l];
               end;
              S:=S-SSM(k+l,TX,TY);
              write( Лицето на обединението е : ,S:5:2);
      END.
```

**11 Задача** Да се намери броя на равностранните триъгълници с различни дължини на основата и върхове от зададеното множество от точки в равнината.

**12 Задача** Дадени са координатите на точките $M_1, M_2, \ldots, M_n$. Намерете триъгълника с максимално (минимално) лице (периметър), с върхове измежду дадените точки.

**Решение:**

Описание на алгоритъма:

За да се намери триъгълника с максимално лице трябва:

1.Да се изберат всички триъгълници,които се образуват от n -те точки.

За това е необходимо да се направи избор на 3 точки от n (реализира се с 3 вложени цикъла).

2.Да се намерят дължините на отсечките, образувани от тези 3 точки.

3.Да се провери могат ли тези три отсечки да образуват триъгълник.

4.Ако се образува триъгълник се намира лицето му .

5.Пази се максималното до момента лице. Отпечатват се координатите на върховете на на търсения триъгълник.

За намиране на тригълника с максимален периметър след намиране на дължините на страните на триъгълника се пресмята периметъра му. Намира се триъгълника с максимален периметър и се отпечатват координатите на върховете му.

```
Program  M4;  { Отпечатва първия намерен триъгълник с това свойство }
Uses Crt,First;
Var      A,B,X,Y : Masiv;
  i1,j1,k1,n,i,j,k:word;
     D1,D2,D3,Max:real;
     BEGIN  { main }
     ClrScr;
      KOORD(n,X,Y);
       Max:=0.0;
        for i:=1 to n-2 do
         for j:=i+1 to n-1 do
          for k:=j+1 to n do
            begin
            D1:=DD(X[i],Y[i],X[j],Y[j]);
            D2:=DD(X[j],Y[j],X[k],Y[k]);
            D3:=DD(X[i],Y[i],X[k],Y[k]);
             if (D1+D2>D3) AND (D1+D3>D2) AND (D2+D3>D1) then
               begin
              A[1]:=X[i];B[1]:=Y[i];
              A[2]:=X[j];B[2]:=Y[j];
              A[3]:=X[k];B[3]:=Y[k];
              A[4]:=A[1];B[4]:=B[1];
              if SSM(3,A,B)>Max then
               begin
                Max:=SSM(3,A,B);
                i1:=i;j1:=j;k1:=k;
               end;
```

```
        end;
      end;
    write( Триъгълникът с максимално лице  ,Max:5:2);
    writeln(  е с върхове с координати: );
    write( ( ,X[k1]:5:2, , ,Y[k1]:5:2, ) ( );
    write(X[j1]:5:2, , ,Y[j1]:5:2, ) ( ,X[i1]:5:2, , ,Y[i1]:5:2, ) );
  END.
```

```
Program  M4B;  { Отпечатва първия намерен триъгълник с това свойство }
Uses Crt,First;
Var  X,Y : Masiv;
  i1,j1,k1,n,i,j,k:word;
    P,D1,D2,D3,Max:real;
      BEGIN  { main }
      ClrScr;
       KOORD(n,X,Y);
        Max:=0.0;
         for i:=1 to n-2 do
          for j:=i+1 to n-1 do
           for k:=j+1 to n do
             begin
             D1:=DD(X[i],Y[i],X[j],Y[j]);
             D2:=DD(X[j],Y[j],X[k],Y[k]);
             D3:=DD(X[i],Y[i],X[k],Y[k]);
              if (D1+D2>D3) AND (D1+D3>D2) AND (D2+D3>D1) then
                begin
                          D2:=DD(X[j],Y[j],X[k],Y[k]);
        D3:=DD(X[i],Y[i],X[k],Y[k]);
         if (D1+D2>D3) AND (D1+D3>D2) AND (D2+D3>D1) then
           begin
             P:=D1+D2+D3;
             if P>Max then
              begin
              Max:=P;i1:=i;j1:=j;k1:=k;
              end;
           end;
         end;
         write( Триъгълникът с максимален периметър  ,Max:5:2);
         writeln(  е с върхове с координати: );
         write( ( ,X[k1]:5:2, , ,Y[k1]:5:2, ) ( );
         write(X[j1]:5:2, , ,Y[j1]:5:2, ) ( ,X[i1]:5:2, , ,Y[i1]:5:2, )
  END.
```

**13 Задача** Изберете четири от n-те точки $M_1, M_2, \ldots, M_n$, които са върхове на квадрат (ромб) с максимален периметър.

**14 Задача** / **обобщение на зад 12 и 13** /

Да се намери k-ъгълника, върховете на който са измежду дадените точки $M_1, M_2, \ldots, M_n$ с минимален (максимален) периметър (лице).

**Решение:**

Описание на алгоритъма:

За да се намери k-ъгълника с максимално лице трябва:

1.Да се изберат всички k-ъгълници. За това е необходимо да се направи избор на k точки от n (комбинации без повторение). В задачата се използва лексикографско пораждане на комбинации.

2.Намира се лицето на всеки k-ъгълник и се пази максималното до момента лице.

Подпрограмата за лице на многоъгълник е реализирана в модула First.

```
Program M5;
Uses Crt,First;
Var     T,S,X,Y : Masiv;
          A : Mas;
     pp,n,i,j,k:word;
          Max:real;
     BEGIN { main }
         ClrScr;Max:=0.0;
          KOORD(n,X,Y);
          repeat
           IIK(3,100,k);
          until k<n;
           { Избор на k елемента от n }
           for i:=1 to k do   A[i]:=i;
           pp:=k; { komb }
            while pp>=1 do
              begin
               { for i:=1 to k do
                           write(A[i],   );}
               for i:=1 to k do
                 begin
            T[i]:=X[a[i]];S[i]:=Y[a[i]];
           end;
            T[k+1]:=T[1];S[k+1]:=s[1];
            writeln(SSM(k,T,S):5:2);  { вс.лица }
          if Max<SSM(k,T,S) then Max:=SSM(k,T,S);
            if A[k]=n then pp:=pp-1
                   else pp:=k;
            if pp>=1 then for i:=k downto pp do
                          a[i]:=a[pp]+i-pp+1;
       end;
       write(k, -ъгълникът с максимално лице   ,Max:5:2);
       writeln(  е с върхове с координати: );
       for i:=1 to k do
```

```
write( ( ,X[A[i]]:5:2, , ,Y[A[i]]:5:2, ) );
END.
```

**15 Задача** Вравнината са зададени **n** точки с координатите си. Да се състави програма, която да определя:

а) ширината на най-тясната полоса, съдържаща точките;

б) центъра на кръг с минимален радиус, съдържащ всички точки.

## 2.3.Общи задачи.

### 2.3.1 Задачи с прави.

**1 Задача** Дадено е множество от точки. Намерете правата, определена от точки от множеството, и съдържаща най-много точки.

**Решение:**
<u>Описание на алгоритъма:</u>

Избираме всевъзможните двойки точки и определяме уравнението на правата, определена от тези две точки. За всяка от останалите точки проверяваме коя лежи на правата (координатите им да удовлетворяват уравнението на правата).

Пазим информация за правата с максимален до момента брой точки.

```
Program K1;
 Uses Crt,First;
  Var    X,Y : Masiv;
         xx,yy :Real;
   Max,i,j,k,n,Br,Im,Jm:word;
      Begin  { main}
      ClrScr;
       Koord(n,X,Y);Max:=0;
        for i:= 1 to n-1 do
          for j:=i+1 to n do
            begin
              Br:=0;
              for k:= 1 to n do
               begin
                if (i<>k)  AND  (j<>k) then
                  begin
                   xx:=X[k]; yy:=Y[k];
                   if (xx-X[i])*(Y[j]-Y[i])-(yy-y[i])*(X[j]-X[i])=0 then
                    Br:=Br+1;
                    end;
                   end;
                   if Br>Max then
                 begin
                   Max:=Br;im:=i;jm:=j;
                           end;
        end;
      write( На правата,определена от точките с номера  ,im,  u  ,jm
      writeln(  лежат най-много точки -  ,Max+2);
      write( Търсената права е определена от точки с координати: );
      write( ( ,X[im]:5:2, , ,Y[im]:5:2, ) u  );
```

```
    write( ( ,X[jm]:5:2, , ,Y[jm]:5:2, ) );
end.
```

Тест на програмата:

Въведете броя на точките : 6
Въведете координатите на точките:
X(1)=2
Y(1)=0
X(2)=1
Y(2)=0
X(3)=0
Y(3)=0
X(4)=1
Y(4)=1
X(5)=2
Y(5)=2
X(6)=3
Y(6)=3
На правата, определена от отчките с номера 3 и 4 лежат най-много точки - 4
Търсената права е определена от точките с координати : (0.00,0.00) и (1.00,1.00).

**2 Задача**. Дадено е множество от прави, зададени с коефициентите на уравненията си.

а)определете броя на пресечните точки на правите;

б)намерете онази права, която има максимален брой пресияания с останалите.

**Решение:**

Описание на алгоритъма:

Вземат се всевъзможните двойки прави. За да се пресичат трябва $b_i.a_j<>b_j.a_i$.
Ако това условие е изпълнено, броят на пресечните точки се увеличава с 1.

За решението на подточка б) за всяка права се намира броя на пресечните и точки с останалите прави. Избира се максималният брой пресечни точки, като се пази най-голямото до момента число.

```
Program K22;  { когато пресечните точки на различните прави не съвпадат }
Uses Crt,First;
Var     A,B,C  :Masiv;
n,i,j,Br,im,b1,Maximum:word;
        Begin
        ClrScr;Br:=0;
        write( n= );read(n);
         for i:= 1 to n do
            begin
             write( A ,i, = );read(A[i]);write( B ,i, = );read(B[i]);
             write( C ,i, = );read(C[i]);
            end;
          Br:=0;
            for i:= 1 to n-1 do
              for j:=i+1 to n do
               if B[i]*A[j]<>B[j]*A[i] then Br:=Br+1;
                writeln( Броят на пресечните точки на правите е  ,Br);
          { б }
               b1:=0;Maximum:=0;
    for i:=1 to n do
      begin
      for j:=1 to n do
       if i<>j then
         if B[i]*A[j]<>B[j]*A[i] then b1:=b1+1;
      if b1>Maximum then begin
         Maximum:=b1;
         im:=i;
         end;
      b1:=0;
     end;
     write( Правата с максимален брой пресичания  , ( ,Maximum, ) );
     writeln(  с останалите е с уравнение: );
     write(A[im]:5:2, .X+ ,B[im]:5:2, .Y+ ,C[im]:5:2, =0 );
   end.
```

<u>Тест на програмата:</u>

n= 4
A1=1
B1=0
C1=0
A2=0
B2=1
C2=0
A3=0
B3=1
C3=2

A4=0
B4=1
C4=3

Броят на пресечните точки на правите е 3.
Правата с максимален брой пресичания ( 3 ) с останалите е с уравнение :
1.00.X+0.00.Y+0.00=0.

**3 Задача** Намерете минималния брой прави, на които могат да се разположат точките от дадено множество.

**4 Задача** Дадено е множество от n точки в равнината, пределени с координатите си. Да се намерят :

а)уравнението на права, определена от две точки от множеството, за която множествата от точки, лежащи от двете ѝ страни се различават с най-малко елементи;

б)уравненията на две прави през две различни точки от множеството, перпендикулярни на свързващата ги отсечка и такава, че полосата, определена от правите, съдържа максимален брой точки от множеството.

**5 Задача** Дадено е множество M от точки в равнината.Да се определи дали за всяка точка A от M съществува точка B от M (A<>B),такава, че не съществуват две точки от множеството M, лежащи от различни страни на правата AB.

**6 Задача** В равнината е зададено множество от по двойки различни прави с коефициентите на уравненията си. Да се посочи тази права, която има максимален брой пресичания с останалите прави.

**7 Задача** Да се намери минималното множество от прави, на които могат да се разположат всички точки от зададено множество от точки в равнината.

**8 Задача /конкурсна/**

В равнината са дадени точките $A, P_1, P_2, \ldots, P_n$ , $n \geq 3$, определени с координатите си. Съставете програма, определяща дали през точката A минава права, разделяща равнината на две полуравнини така, че всички останали точки $P_1, P_2, \ldots, P_n$ да се намират само в едната от тях.

**9 Задача /Конкурсна/ сп."Компютър" бр.3 /1994 г.**

В равнината са дадени няколко отсечки с кординатите на краищата си. Да се състави програма за намиране на права линия,която пресича всяка от дадените отсечки.

**10 Задача /Конкурсна/ сп."Математика" бр.5 /1989 г.**

Дадени са 2n ( $n \leq 12$ ) точки, номерирани с числата от 1 до 2n, никои три от които не лежат на една права. Да се състави програма за съединяване на точките с отсечки, така, че:

- всяка точка да е свързана с точно n други;

-от всяка точка може да се достигне до всяка друга точка, като се премине по не повече от n отсечки.

Да се докаже, че при такъв начин на свързване съществува затворен маршрут от 2n отсечки, който минава през всички дадени точки. Програмата да отпечатва един такъв маршрут, както и съобщение с кои n на брой точки е свързана всяка от точките 1,2,. . .,2n.

## 2.3.2 Задачи за окръжности.

**1 Задача** Да се намери минималното множство от окръжности на които могат да се разположат всички точки от дадено множестчо от точки в равнината, зададени с координатите си.

**2 Задача** Дадено е множество M от n точки в равнината, определени с координатите си.

а)Да се построи окръжност така, че разликата между броя на точките вътре и вън от съответния кръг да бъде минимална. Задчата да се реши и когато точките през които минава търсената окръжност принадлежат на множеството.

б)Да се определят центъра и радиуса на окръжността, съдържаща най-много точки от множеството. Окръжността да е определена от точки от множеството.

в)Да се определи окръжността, минаваща през три различни точки от множеството, съответния кръг на която съдържа максимален брой точки от множеството.

**Решение на б)**

Описание на алгоритъма:

Избираме всевъзможните тройки точки (3 вложени цикъла). Всяка такава тройка определя по една окръжност.

Намираме центъра и радиуса на всяка такава окръжност по следния начин :

1.Нека окръжността да е определена от точките с номера на координатите i, j , k.

Образуваме отсечките определени от точките i ,j и j,k .

2.Намираме уравненията на симетралите на тези отсечки.

3.Центъра на търсената окръжност е пресечната им точка O. Намираме координатите на пресечната им точка.

4.Радиусът е равен на разстоянието между точката O и например точката i.

5.За всяка такава окръжност определяме броя на точките, които лежат върху нея.

```
Program K1BB;
Uses Crt,First;
 Var                              X,Y : Masiv;
eps,x00,y00,rr, x0,y0,t,s,p,q,r,u,Radius :Real;
          m,Max,i,j,k,n,Br,Im,Jm,km:word;
     Begin  { main}
     ClrScr; write(sqrt(3));
      Koord(n,X,Y);Max:=0;Br:=0;
      write( eps= );read(eps);
       for i:= 1 to n-2 do
        for j:=i+1 to n-1 do
         for k:=j+1 to n do
          begin
           Br:=0;
           P:=2*(X[j]-X[i]);Q:=2*(Y[j]-Y[i]);
           R:=SQR(X[i])+SQR(Y[i])-SQR(X[j])-SQR(Y[j]);
           S:=2*(X[k]-X[j]);T:=2*(Y[k]-Y[j]);
           U:=SQR(X[j])+SQR(Y[j])-SQR(x[k])-SQR(Y[k]);
                  if (P<>0.0) AND (T*P-S*Q<>0.0) then
        begin
          x0:=(U*Q-R*T)/(T*P-S*Q);
          y0:=(S*R-P*U)/(T*P-S*Q);
          Radius:=DD(X0,X[i],Y0,Y[i]);
          for M:=1 to N do
           if (M<>i) AND (M<>j) AND (M<>k) then
           if ABS((X[M]-X0)*(X[M]-X0)+(Y[M]-Y0)*(Y[M]-Y0)-SQR(Radiu
           { с точност eps }
               then  Br:=Br+1;
          if Br>Max then begin
                    Max:=Br;rr:=radius;x00:=x0;y00:=y0;
                    im:=i;jm:=j;km:=k;
                    end;
      end;
    end;
  write( Окръжността,съдържаща най-много точки е определена от  );
  writeln( точките с координати: );
       write ( ( ,X[im]:5:2, , ,Y[im]:5:2, )  ( );
  write(X[jm]:5:2, , );
  writeln(Y[jm]:5:2, ) и ( ,X[km]:5:2, , ,Y[km]:5:2, ) );
  writeln( Броят на точките,лежащи на тази окръжност е  ,Max+3);
  write( Радиусът и е = ,rr:5:2,  а центърът и е с координати  ( );
  write(x00:5:2, , ,y00:5:2, ) );
end.
```

Резултати от изпълнението на програмата:

Въведете броя на точките: 8
Въведете координатите на точките:
X(1)=2
Y(1)=0
X(2)=0
Y(2)=-2
X(3)=-2
Y(3)=0
X(4)=0
Y(4)=2
X(5)=1
Y(5)=1.7320508076
Eps=0.001
Окръжността,съдържаща най-много точки е определена от точките с координати:
(2.00,0.00) (0.00,-2.00) и (-2.00,0.00).
Броят на точките, лежащи на тази окръжност е 5 .
Радиусът и е = 2.00 , а центърът и е с координати (0.00,0.00).

**3 Задача** Дадено е множество от окръжности (Ci,ri). Окръжностите A и B се наричат свързани, ако се пресичат или има трета окръжност, която ги пресича. Изберате максималното подмножество на две по две несвързани една с друга окръжности.

**Решение:**

Описание на алгоритъма:

За да са свързани окръжностите A и B те трябва или да се пресичат , или да има трета окръжност, която да ги пресича.

Ako $C_i$ и $C_j$ са центровете на две окръжности трябва да се провери дали разстоянието $d_1$ между тях е по-малко от $r_i+r_j$ ( допирането не се включва ).

Ako $d_1<r_i+r_j$ тогава окръжностите са свързани, иначе се търсят разстоянията $d_2$ между $C_i$ и $C_k$ и $d_3$ между $C_j$ и $C_k$.

Ako $d_2<r_i+r_k$ или $d_3<r_j+r_k$ , то третата окръжност пресича една от двете и следователно окръжностите i и j са свързани.

Функцията Swarz проверява дали окръжностите i и j са свързани. Тя е логическа функция и приема стойност true или false, взависимост от това дали условието е изпълнено.

В главната програма се броят двойките несвързани окръжности.

Реализирано е с два вложени цикъла.

```
Program K7;
Uses Crt,First;
 Var   R,X,Y  :Masiv;
         KK  :Mas;
flag,n,i,j,k   :word;
  function SWARZ(n,i,j,k :word;Var R,X,Y:Masiv;Var Flag:word):boolean;
    begin
    flag:=0;
     if DD(X[i],Y[i],X[j],Y[j])<R[i]+R[j] then SWARZ:=true
     else
     if (DD(X[i],Y[i],X[k],Y[k])<R[i]+R[k]) AND (DD(X[j],Y[j],X[k],Y[k])
     <R[j]+R[k]) then
      begin
       Swarz:=true;flag:=1;
      end
       else
         Swarz:=false;
   end;
  Begin  { main}
 ClrScr;
 writeln( Въведете броя и координатите на центровете на окръжностите );
 Koord(n,X,Y);
 writeln( Въведете радиусите на окръжностите : );
 for i:=1 to n do begin write( R( ,i, )= );read(R[i]);end;
 for i:=1 to n do  KK[i]:=1;
  for i:=1 to n-1 do
   for j:=i+1 to n do
    for k:=1 to n do
     begin
      if (KK[i]=1) AND (KK[j]=1) then if KK[k]=1 then
       if Swarz(n,i,j,k,R,X,Y,Flag)=true then
         begin
         if flag=1 then
           begin
            KK[i]:=0;KK[j]:=0
           end
                 else
            begin
             KK[i]:=0;KK[j]:=0;KK[k]:=0;
            end;
        end;
   end;
   write( Номерата на окръжностите,участвуващи в максималното );
   writeln(  подмножество са: );
   for i:=1 to n do
   if KK[i]=1 then writeln( номер  ,i,  с радиус  ,R[i]:5:2);
end.
```

Тест на програмата:

Въведете броя и координатите на центровете на окръжностите.
Въведете броя на точките: 5
Въведете координатите на точките:
X(1)=0
Y(1)=0
X(2)=0
Y(2)=-4
X(3)=4
Y(3)=0
X(4)=0
Y(4)=-1
X(5)=0
Y(5)=1
Въведете радиусите на окръжностите :
R(1)=1
R(2)=1
R(3)=2
R(4)=1
R(5)=1
Номерата на окръжностите, участвуващи в максималното подмножество са:
номер 2 с радиус 1.00
номер 3 с радиус 2.00
номер 5 с радиус 1.00

**4 Задача** От дадено множество от точки в равнината да се изберат две различни точки така, че окръжностите с даден радиус и центрове в тези точки да съдържат вътре в себе си еднакъв брой точки.

**Решение:**
Описание на алгоритъма:

1.Въвеждат се броя и координатите на точките.
2.Въвежда се радиуса R.
3.За всяка точка :
- броим точките, които са вътрешни за окръжността (принадлежат на кръга, ограничен от окръжността) с радиус R и център тази точка (намираме разстоянията между точката, която се явява център на окръжността и всяка друга точка и броим онези, които са по-малки от R и ги записваме в масив BR );
- търсим първите две точки, които са центрове на окръжности ,в чиито кръг се съдържат равен брой точки, т.е. търсим първите два равни елемента на масива BR.

```
Program O4;
Uses Crt,First;
 Var   BR,X,Y : Masiv;
          R :Real;
     i,j,n,B :word;
      Begin  { main}
       ClrScr;
       Koord(n,X,Y);
       write( R= );read(R);
      for i:= 1 to n do
       begin
        b:=0;
         for j:=1 to n do
          if i<>j then
            if DD(X[i],Y[i],X[j],Y[j])<R then B:=B+1;
            BR[i]:=B;
      end;
         for i:= 1 to n do
                    for j:=i+1 to n do
        if BR[i]=BR[j] then begin
         write( Търсените точки са с координати );
         write(  ( ,X[i]:5:2, , ,Y[i]:5:2, ) и ( );
         write(X[j]:5:2, , ,Y[j]:5:2, ) );
         exit;end;
        write( Няма точки с такова свойство. );
 end.
```

<u>Тест на програмата</u>:

Въведете броя на точките : 7
Въведете координатите на точките :
X(1)=0
Y(1)=0
X(2)=1
Y(2)=-1
X(3)=1
Y(3)=1
X(4)=5
Y(4)=0
X(5)=6
Y(5)=0
X(6)=5
Y(6)=1
X(7)=8
Y(7)=0
Търсените точки са с координати: (0.00,0.00) и (5.00,0.00)

**5 Задача** Даден е масив от точки $[x_i,y_i]$ , $i=1,2,\ldots,100$ и числата $R_1$ и $R_2$. Да се състави програма за определяне на точките $(x_i,y_i)$, лежащи вътре в пръстена, образуван от окръжностите с център точката с максимално x и радиуси $R_1$ и $R_2$.

**6 Задача** В равнината са зададени n точки с координатите си $A_i(x_i,y_i)$ $i=1,2,\ldots,n$. Да се състави програма, която определя минималния брой окръжности, инцидентни с всички точки.

**7 Задача /Конкурсна сп."Компютър" бр.2 1993 г. /**

Дадени са координатите на n точки в равнината. Да се състави програма за построяване на не повече от n окръжности, така че да са изпълнени следните три условия:

1.Всяка от дадените точки да е вътре в някоя от построените окръжности.

2.Всеки две от окръжностите да са отдалечени поне на разстояние една мерна единица.

3.Сумата от диаметрите на всички построени окръжности да е по-малка от n мерни единици.

**8 Задача /Конкурсна сп."Математика" бр.8 1989 г. сп."Математика" бр.7 1990 г. /**

В равнината са дадени n точки, $(4 \le n \le 100)$ с координатите си, които са цели числа в интервала [-100,100]. Да се състави програма за намиране на три от дадените точки (ако такива съществуват) през които минава окръжност със свойството :

а)вън от нея не лежи никоя от дадените точки;

б)вътре в кръга, ограничен от окръжността, не лежи никоя от дадените точки.

**9 Задача /Конкурсна сп."Математика +" бр.3 1994 г. /**

Да се напише програма,която при въведено естествено число n :

а)построява в равнината n окъжности, всяка от които пресича всяка от останалите точно в две точки;

б)ако броят на всички пресечни точки е k ,програмата да посочи еднозначно-обратимосъответствие между множеството на пресечните точки и множеството от целите числа в интервала [1,2,. . .,k] със следното свойство:

- сумата от съпоставените числа на пресечните точки, принадлежащи на коя да е окръжност, да е равна на сумата от съпоставените числа на пресечните точки от коя да е друга окръжност.

## 2.3.3 Задачи за многоъгълници.

### 2.3.3.1 Триъгълници.

**1 Задача** В равнината е зададено множество от n произволно пресичащи се отсечки с координатите на краищата си. Да се преброят всички триъгълници, образувани от отсечките.

**2 Задача** Дадено е множество М от n точки в равнината, определени с координатите си. Да се изберат три различни точки от множеството, така че вътре в триъгълника с върхове тези точки да се съдържат максимален брой точки от множеството М.

**3 Задача** Намерете броя на всички остроъгълни триъгълници с върхове от зададено множество от n точки в равнината. Да се построи това множество от триъгълници.

**Решение:**

Описание на алгоритъма:

За решението на задачата се изисква :

1. Да се изберат всевъзможните тройки от n точки.
2. За всяка такава тройка точки да се провери:
   - дали отсечките, образувани от тези точки могат да бъдат страни на триъгълник;
   - дали cos a , cos b ,cos c са в интервала [0,1] (за да бъде триъгълника остроъгълен ).
3. Да се преброят онези триъгълници, които отговарят на тези условия.

Косинусите на ъглите на триъгълниците се изчисляват по косинусовата теорема, чрез страните на триъгълниците.

```
Program K6;
 Uses Crt,First;
  Var      X,Y : Masiv;
  ca,cb,cg,a,b,c :Real;
        i,j,k,n,Br:word;
      Begin  { main}
      ClrScr;
       Koord(n,X,Y);Br:=0;
        for i:= 1 to n-2 do
          for j:=i+1 to n-1 do
           for k:=j+1 to n do
            begin
             a:=DD(X[i],Y[i],X[j],Y[j]);
             b:=DD(X[j],Y[j],X[k],Y[k]);
             c:=DD(X[i],Y[i],X[k],Y[k]);
             CA:=(b*b+c*c-a*a)/2*b*c;
             CB:=(a*a+c*c-b*b)/2*a*c;
             CG:=(a*a+b*b-c*c)/2*a*b;
             {[writeln( cos= ,ca:5:2,   ,cb:5:2,   ,cg:5:2);
          writeln(a:5:2,   ,b:5:2,   ,c:5:2);}
          if (a+b>c) AND (b+c>a) AND (c+a>b) then
          if (0<ca) and (ca<1) and (0<cb) and (cb<1) and (0<cg) and
           (cg<1) then  Br:=Br+1;
         end;
    write( Броят на остроъгълните триъгълници е  ,Br);
   end.
```

Тест на програмата:

Въведете броя на точките: 5
Въведете координатите на точките:
X(1)=0
Y(1)=0
X(2)=1
Y(2)=0
X(3)=1
Y(3)=1
X(4)=0.5
Y(4)=1
X(5)=0
Y(5)=1

Броят на остроъгълните триъгълници е 1.

**4 Задача** В равнината е дадено множество от n точки ( n ≥ 3). Между някои двойки точки са прекарани отсечки. Разстоянията между двойките точки са зададени в матрицата D(n,n). Елементите D(i,i) ,1 ≤ i ≤ n по главния диагонал са нули. Ако между точките i и j няма отсечка, то елементът D(i,j)=-1, а ако има, елементът D(i,j) е равен на дължината на отсечката. Съставете програма, която пресмята броя на триъгълниците с върхове от множеството от n точки и със страни от зададеното множество от отсечки.

**Решение:**

Описание на алгоритъма:

1.Въвеждаме елементите на матрицата, които са разположени под главния диагонал.

2.Попълваме симетричните елементи със същите стойности.

3.Избираме всевъзможните тройки от n точки (с три вложени цикъла ).

4.За всяка такава тройка вземаме разстоянията между точките от матрицата D и проверяваме тези отсечки могат ли да бъдат дължини на страни на триъгълник.

5.Преброяваме тези, които отговарят на това условие.

```
Program K8;
 Uses Crt,First;
 type dd=array [0..30,0..30] of real;
  Var    D:dd;
     i,j,k,n,Br:word;
       Begin  { main}
       ClrScr;
       write( n= );read(n);writeln;
       write( Въведете матрицата : ); { под гл.диаг. се въвежда }
       for i:= 2 to n do
         for j:=1 to i-1 do
         begin
```

```
      write( Въведете D( ,i, , ,j, )= );read(D[i,j]);
      D[j,i]:=D[i,j];
    end;
    for i:=1 to n do
     D[i,i]:=0;
      Br:=0;
     for i:=1 to n-2 do
       for j:=i+1 to n-1 do
         for k:=j+1 to n do
           if (D[i,j]+D[j,k]>D[i,k]) AND (D[j,k]+D[i,k]>D[i,j]) AND
           (D[i,j]+D[i,k]>D[j,k]) then Br:=Br+1;
    write( Броят на триъгълниците е  ,Br);
end.
```

<u>Тест на програмата</u> :

N=4
Въведете матрицата:
Въведете D(2,1)=2
D(3,1)=2
D(3,2)=2
D(4,1)=1
D(4,2)=1
D(4,3)=2
Броят на триъгълниците е 3.

**5 Задача** Да се изберат три различни точки от зададено множество от точки в равнината, така че разликата между броя на точките вътре и вън от триъгълника с върхове избраните точки да е минимална.

**6 Задача** От триъгтълници с върхове от зададено множество от точки в равнината да се избере такъв, чиито страни да съдържат максимален брой точки от даденото множество.

**7 Задача / Конкурсна сп."математика" бр.1 1990 г. /**

Дадени са n (3<n<100) точки, разположени в равнината и номерирани с естествените числа от 1 до n. Някои от точките са свързани с отсечки. Информацията за отсечките (номер на началната и номер на крайната точка) се въвежда от клавиатурата. Като признак за край на входните данни да се използва "отсечката" (0,0). Да се състави програма,която:

а)прочита информацията за отсечките, проверява нейната коректност и отпечатва входните данни;

б)отпечатва всички триъгълници, образувани от дадените отсечки.

**8 Задача** Да се построят два триъгълника с върхове от зададено множество от точки в равнината, така че първия триъгълник да лежи изцяло във втория.

### 2.3.3.2 Четириъгълници.

**1 Задача** Да се построи множеството от всички различни изпъкнали четириъгълници с върхове от зададено множества от точки в равнината.

**2 Задача** Да се намери положението в равнината на правоъгълника със зададени дължини на страните, при условие, че върховете му трябва да имат целочислени координати и вътре в него трябва да се намират максимален брой точки от даденото множество.

**3 Задача /Конкурсна сп. "Компютър" бр.1 1989 г. /**
Даден е списък от правоъгълници в равнината, зададени с координатите на върховете си. Страните им са успоредни на координатните оси. Да се състави програма, която отпечатва координатите на точка, принадлежаща едновременно на всички правоъгълници, ако такава съществува.

**4 Задача / Конкурс Слънчев бряг 1983 г. /**

Дадена е правоъгълна координатна система, точка M със своите координати и редицата от квадрати $K_1, C_1, K_2, C_2, \ldots, K_{n-1}, C_{n-1}, K_n$. Върховете на квадрата $K_i$ $i=1,2,\ldots,n$ са с координати : $(a_i,a_i)$ , $(-a_i,a_i)$ , $(-a_i,-a_i)$ , $(a_i,-a_i)$ , където $a_1=1$ , $a_{i+1}=2a_{i+1}$ , $i=1,2,\ldots,n-1$.

Страните на квадрата $C_i$ $i=1, 2, \ldots, n-1$ съдържат върховете на квадрата $K_i$ и са успоредни на диагоналите на $K_i$.

а) Съставете програма, чрез която да се определя номера на квадрата с най-малко лице, съдържащ точката M, или да се изведе съобщение, че точката не се съдържа в нито един от квадратите;

б)определя максималният брой квадрати, които трябва да бъдат разгледани от предложения от вас алгоритъм за намиране на търсения квадрат.

### 2.3.3.3Многоъгълници.Вътрешна точка за многоъгълник. Разделяне многоъгълника на триъгълници.

**1 Задача** Да се състави програма, която по зададено n определя координатите на върховете на правилния n-ъгълник със страни 1, център (0,0) и първи врьх по оста Ox, а също така и координатите на върховете на всички триъгълници, които се получават от пресичането на диагоналите на n-ъгълника.

**2 Задача** Точка P е вътрешна за даден изпъкнал n-ъгълник, $n \geq 5$. Да се състави алгоритъм за намиране върховете на друг изпъкнал многоъгълник, който е с максимално лице, различен е от дадения, съдържа точката P, а върховете му са върхове на дадения многоъгълник.

**3 Задача** Дадени са n точки в равнината с координати $A_i(x_i,y_i)$ , $i=1,2,\ldots,n$.
Да се състави програма, която определя:

а)координатите на върховете на многоъгълника с минимален периметър, съдържащ всички точки;

б)изпъкналия многоъгълник с минимално лице, съдържащ всички точки.

**4 Задача** В равнината са дадени точката A(x,y) и многоъгълник M с върхове $M_i(x_i,y_i)$ , i=1,2,. . .,n. Да се намери разстоянието между A и M.

**5 Задача** Нека M е прост многоъгълник, зададен с координатите на върховете си. Точките A(x,y) и B(z,t) са два от върховете му. Намерете координатите на върховете на минималната по дължина начупена отсечка с начало в точката A и край B, изцяло съдържаща се в M.

**6 Задача** Простите непресичащи се многоъгълници $M_1,M_2,. . . ,M_s$ са зададени в равнината с координатите на върховете си. Нека точките $X(x_1,y_1)$ и $У(y_1,y_2)$ са външни за всички многоъгълници. Да се състави програма за :

а)въвеждане на коректни данни;

б)печат на координатите на върховете на начупената отсечка с краища точките X и У, непресичаща никой от многоъгълниците $M_i$, i=1,2, . . ,s;

в)като б) и такава, че разликата от броя многоъгълници от двете и страни да е минимална по абсолютна стойност.

## Вътрешна точка за многоъгълник

За решаването на следващите задачи е необходимо да се разгледа [5].

**1 задача /Конкурсна сп."Компютър" бр.12 1991 г.**

В равнината е зададен прост, но не непременно изпъкнал, многоъгълник с координатите на върховете си. Върховете са номерирани последователно според избраната посока на обхождане на контура на многоъгълника. Съставете програма, която да определя дали дадена точка T(x,y) е вътрешна за многоъгълника.

**2 Задача / Сп."Обучението по математика" бр.4 1985 г. /**

Точката P е от равнината на четириъгълника с върхове A,B,C,D, всички зададени с координатите си. Да се състави програма, която определя дали точката P е вътрешна или не за четириъгълника в случаите, когато той е :

а)квадрат;

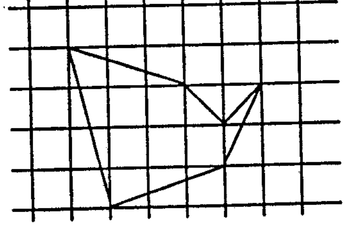
б)ромб;

в)изпъкнал четириъгълник;

г)неизпъкнал прост четириъгълник.

**3 Задача /конкурсна сп."Математика +" бр.1 1993 г. /**

Дадена е квадратна решетка от перпендикулярни прави в равнината и многоъгълник с върхове, лежащи върху възлите на решетката, както е показано на фигурата.



Да се състави програма, която:

а)намира броя на възлите от решетката, които са вътрешни за многоъгълника;

б)намира броя на възлите от решетката, които са върху контура на многоъгълника;

в)пресмята лицето на многоъгълника.

**4 Задача /Конкурсна сп."Обучението по математика и информатика" бр.4 1994 г./**

Дворно място с формата на многоъгълник с n страни $(3 \le n \le 50)$ е оградено с висока, плътна (непрозрачна) ограда. Върховете на многоъгълника са определени с координатите си. P е точка от равнината на многоъгълника, определена също с координатите си.

а)Да се определи площта на дворното място и дължината на оградата;

б)Да се определи дали точката е вътрешна за многоъгълника;

в)Ако точката P е вътрешна за многоъгълника да се определят онези негови върхове, които са видими от точката P.

Входните данни се четат от текстов файл, чието име се въвежда от клавиатурата. Първият ред на файла съдържа координатите на точката P, а останалите редове съдържат координатите на последователните върхове на многоъгълника. Резултатите се извеждат на екрана.

## Разделяне на многоъгълника на триъгълници

При решаването на следващите задачи е необходимо да се разгледа [2].

**5 Задача** Даден е изпъкналият N-ъгълник $M=M_1M_2 . . .M_n$. Той може да се раздели по различни начини на триъгълници с непресичащи се диагонали. Да се намери:

а)броят на разделянията на многоъгълника на триъгълници с непресичащи се вътрешниости;

б)да се намери делението, при което сумата от дължините на диагоналите да е минимална (с не повече от $O(n^3)$ операции).

## 2.3.4.Задачи с различни геометрични фигури

**1 Задача** В равнината е зададена окръжност с център (x,y) и радиус R , x,y,R са реални числа. Да се определи колко целочислени единични квадрати лежат вътре в окръжността.

**2 Задача** От зададено множество от точки да се изберат три различни точки, така че разликата между лицето на кръга, ограничен от окръжността, минаваща през тези три точки и лицето на триъгълника с върхове тези точки да е минимална.

**3 Задача** В равнината са зададени множество от точки M и кръг. Да се изберат от M две различни точки, така че минимално да се различава броя на точките в кръга, лежащи от различни страни на правата, определена от тези две точки.

**4 Задача** Да се построят **n** прави, така че на всяка права да лежат две различни точки от зададено множество от точки в равнината, а броят на триъгълниците, образувани при пресичането на правите да е максимален.

**5 Задача / Конкурсна сп."Компютър" бр.7 1995 г. /**
Разглеждаме точките в равнината, които имат целочислени координати и се намират вътре в даден правоъгълник с размери **а** на **в** единици. Напишете програма, която определя броя на всички нееднакви помежду си триъгълници, чиито върхове съвпадат с с някои от разглежданите точки.

**6 Задача** В равнината е зададена окръжност с център (x,y) и радиус R, където x,y, R са реални числа. Да се определи колко целочислени единични квадрати лежат вътре в окръжността

**7 Задача /Конкурсна сп."Математика" бр.7 1986 г. /**
В квадрата OABC с размери 70×70 са разположени 5 фигури - 2 кръга и 3 правоъгълника, никои две от които не се припокриват. Известно е, че двата кръга имат радиуси, равни на 5. Размерите на правоъгълниците са 15×10 , 25×15 , 30×30 и страните им са успоредни на страните на квадрата OABC.

а)Да се докаже, че в квадрата OABC може да се разположи още един кръг с радиус 5, така че да не се припокрива с никоя от дадените 5 фигури;

б)Да се намери алгоритъм за разполагане на кръга от а).

## 2.3.5 Пространствени задачи.Център на тежестта.

## Център на тежестта

Множество от материални точки в тримерното пространство е такова крайно множество от точки, за всяка от които е указана нейната маса - положително число ([3]).

Нека са дадени **n** точки $A_1, A_2, \ldots, A_n$ . На всяка от тях съпоставяме по едно реално число : $m_1$ на $A_1$ , $m_2$ на $A_2$ ,. . . ,$m_n$ на $A_n$ , като $m_1+m_2+\ldots+m_n \neq 0$. Тези числа се наричат маси. Ще казваме, че е дадена система от материални точки $A_1, A_2, \ldots, A_n$, съответно с маси $m_1, m_2, \ldots, m_n$.

Точката G се нарича център на тежестта на тази система,ако за нея е изпълнено векторното равенство

$$m_1\vec{GA_1}+m_2\vec{GA_2}+\ldots+m_n\vec{GA_n}=0 .$$

Всяка крайна система от материални точки има точно един център на тежестта.

Центърът на тежестта на множество от материални точки Pi(xi,yi,zi) с маси $m_i$ има координати :

$$x_c=\frac{\Sigma x_i m_i}{\Sigma m_i} \qquad y_c=\frac{\Sigma y_i m_i}{\Sigma m_i} \qquad z_c=\frac{\Sigma z_i m_i}{\Sigma m_i}$$

**1 Задача** В тримерното пространство е зададено множество от материални точки. Да се намери онази точка, която е разположена най-близо до центъра на тежестата на това множество.

**2 Задача** В тримерното пространство е зададено множество от материални точки. Всяка от точките с максимална маса изчезва, губейки 1/10 част от масата си и раздавайки останалата си маса по равно на всички останали по-леки точки. Да се определи сумарната маса на множеството от материални точки в този момент, когато всички останали точки в нея имат еднаква маса.

**3 Задача** В тримерното пространство е зададено множество от материални точки. Да се намери разбиването на това множество на две непразни и непресичащи се подмножества, така че техните центрове на тежестта да са най-близко един до друг.

**4 Задача** В условието на предната задача да се намери такова подмножество, съдържащо точно n материални точки, центърът на тежестта на което да е най-близо до началото на координатната система.

## В пространството

**5 Задача** Дадени са координатите на n точки. Да се отпечатат тези от тях, които принадлежат на:

а)сферата $x^2+y^2+z^2<=R^2$;

б)полупространството $ax+by+cz<=d$;

в)сечението на сферата и полупространството.

**6 Задача** Многоъгълник в пространството е зададен с координатите на върховете си. Да се определят всички върхове на многоъгълника, двойката страни на който сключват прав ъгъл.

**7 Задача** Многостенът $M=A_1A_2...A_n$ е зададен с координатите на върховете си $A_i(x_i,y_i,z_i)$ ,i=1,2,. . . ,n. Да се състави програма, която:

а)въвежда n , $(x_i,y_i,z_i)$ , i=1,2,. . . ,n и проверява дали многостенът е правилен;

б)с не повече от $O(n^3)$ операции определя броя разделяния на M (чрез секущи равнини, минаващи през върховете и страните на M) на многостени с непресичащи се вътрешности и това от разделянията, за което площта на разделящите равнини е минимална.

**8 Задача** дадени са M точки в пространството,определени чрез координатите си. Да се състави програма за определянето на :

а)кълбо с даден радиус R , което съдържа максимален брой от точките;

б)координатите на върховете на всички изпъкнали четириъгълници с върхове дадените точки (разглежданията да се обобщят за изпъкнали k-ъгълници);

в)всички тройки от точки, лежащи на една права;

г)максималният брой от точки, лежащи на една права (окръжност, в една равнина);

д)минималният и максималният радиуси на окръжности, минаващи през 3 от точките;

е)най-тясната ивица, образувана две успоредни равнини, съдържаща всички точки;

ж)триъгълника с върхове три от точките и с минимало лице.

**9 Задача** Даден е паралелепипед, съставен от $n \times r \times s$ кубчета, оцветени в един от цветовете $C_1, C_2, \ldots, C_k$. Две кубчета се наричат съседни, ако имат обща стена. Две едноцветни кубчета се наричат свързани, ако са съседни или съществува редица от съседни кубчета от същия цвят, като първото кубче е съседно с първия член на редицата, а второто - с последния. Тяло се нарича множество от поне две едноцветни кубчета в което всеки две кубчета са съседни. Да се състави програма за определяне като въведете подходящо кодиране.

**10 Задача** Зададено е множество M от точки в тримерното пространство. Да се намери такава точка, че кълбото с даден радиус и център в тази точка да съдържа максимален брой точки от M.

**11 Задача** Множеството от по двойки различни равнини в тримерното пространство е зададено с изброяването на тройките точки, през които минава всяка от равнините. Да се избере максималното подмножество от по двойки неуспоредни равнини.

**12 Задача** Зададено е множество от точки в тримерното пространство. Да се намери минималния радиус от радиусите на кълбата с центрове в тези точки, съдържащи точно n точки от това множество.

## 2.3.6 Множества от точки.Подмножества.Операции с множества (сечение,разлика,обединение,покриване на множества от точки).

**1 Задача** Дадени са две множества от точки в равнината.

а)да се изберат три различни точки от първото множество, така че:

- триъгълника с върхове в тези точки
- кръга, ограничен от окръжността, минаваща през три различни точки да съдържа (покрива) всички точки от второто множество и да има минимално лице;

б)да се изберат 4 различни точки от първото множество, така че квадрата с върхове тези точки да съдържа всички точки от второто множество и да има минимално лице.

**2 Задача** N - ъгълник в равнината е зададен с координатите на върховете си $A_i(x_i,y_i)$ , $i=1,2,\ldots,n$. Да се състави програма, която намира представяне на многоъгълника като обединение на :

а)изпъкнали многоъгълници с непресичащи се вътрешности;

б)минимален броъй многоъгълници от а).

**3 Задача** Два изпъкнали многоъгълника са зададени с координатите на

**4 Задача** Ще определим ред на точките в равнината по следния начин: $(x,y) \le (u,v)$ ако, или $x<u$, или $x=u$, или $y \le v$. Да се преброят точките от даденото множества от точки в равнината в съответствие с този ред.

**5 Задача** Дадени са две множества от точки в равнината. Да се построят сечението и разликата на тези множества.

**6 Задача** Множество от точки в равнионата ще наричаме "редовно", ако заедно с всяка двойка различни точки, то съдържа още една трета - върха на правилен триъгълник с върхове в тези точки. Да се определи "редовно" ли е зададено множество от точки.

**7 Задача** В равнината са зададени **n** множества от точки. Във всяко множество има **m** точки. Сред точките от първото множество да се намери такава, която принадлежи на най-голям брой множества.

**8 Задача** В равнината са зададени множество от точки A и множество от окръжности B. Да се намерят две различни точки от A, такива че правата определена от тях се пресича с максимален брой окръжности от B.

**9 Задача** В равнината са зададени множество от точки A и множество от прави B. Да се намерят две различни точки от A, такива че правата, определена от тях да е успоредна на най-голям брой прави от B.

**10 Задача** Дадени са 3n точки в равнината, при което никои 3 от тях не лежат на една права. Да се построи множеството от триъгълници, с върхове в тези точки, така че никои два триъгълника не се пресичат и не се съдържат един в друг.

**11 Задача** Дадени са две непресичащи се крайни множества от точки в равнината. Да се определи окръжността, минаваща през k ($k \ge 3$) точки на всяко от множествата.

**12 Задача** Дадени са две множества от точки в равнината. От първото множество да се изберат 3 различни точки, така че триъгълника с върхове в тези точки:

а)да съдържа строго вътре в себе си равен брой точки от двете множества;

б)да покрива всички точки от второто множество и да има минимално лице.

**13 Задача** Дадени са две множества от точки в равнината. Да се намерят центъра и радиуса на окръжността, минаваща през k ($k \ge 3$) точки от първото множество и съдържаща строго в себе си :

а) **m** точки от второто множество;

б)равен брой точки от първото и второто множество.

**14 Задача** Дадени са две множества от точки в равнината. Да се изберат четири различни точки от от първото множество, така че квадратът с върхове в тези точки да покрива всички точки от второто множество и да има минимално лице.

**15 Задача** Дадени са две множеаства от точки в равнината. Да се изберат три различни точки от първото множество, така че кръга, ограничен от окръжността, минаваща през тези три точки да съдържа всички точки на второто множество и да има минимално лице.

**16 Задача** Зададено е множество от точки в равнината, нележащи на една права. Да се определи минималното подмножество от точки, след отделянето на които остават точки, лежащи на една права.

**17 Задача** В равнината е зададено множество от точки и окръжност с радиус R и и център в началото на координатната сиатема. Да се построи множеството от всички триъгълници с върхове от зададените точки, имащи непразно сечение с окръжността.

**18 Задача** Дадено е множество от точки в равнината.Да се построят всички максимални подмножества от точки, лежащи на една права, които съдържат повече от две точки.

**19 Задача** Даени са две множества, всяко съдържащо **n** точки в равнината.
Точките са зададени с координатите си. Да се определи дали двете множества са еднакви в смисъл на обикновената (евклидовата) геометрия.

**2.3.7. Конкурсни задачи.**

1 Задача / сп."Обучението по математика и информатика" бр.3 1987 г. /

**I кръг на олимпиадата по информатика за 8-11 клас II 1987 г.**

Да се състави програма, която последователно извършва следното:

а)въвежда координатите на **n** точки ($n \geq 1$) в равнината $A_1, A_2, \dots, A_n$;

б)изобразява върху екрана на микрокомпютъра само видимата част на начупената линия Ф, съставена от отсечки, съединяващи последователно точките $A_1, A_2, \dots, A_n$. (Ако т. $A_i$ например е вън от графичното поле на микро-компютъра, тогава отсечките $A_{i-1}A_i$ и $A_iA_{i+1}$ не трябва да се изчертават.);

в)обръща се към подпрограмите, описани в задачи 2 и 3.

*Задача 2.* Нека P е правоъгълник, зададен с координатите си и със страни, успоредни на координатните оси (на координатната система на графичното поле на микрокомпютъра).

а)да се определи възможно ли е цялата фигура Ф да се премести във вътрешността на правоъгълника P, използувайки движениятя, описани в задача 3;

б)да се определи дали съществува поне една четворка от последователни точки $A_i, A_{i+1}, A_{i+2}, A_{i+3}$ ($i=1,2,\dots,n$) измежду определените по-горе, които да са последователни върхове на правоъгълен трапец.

*Задача 3.* Да се опише чрез подпрограма придвижването на фигурата Ф в четирите посоки (нагоре,надолу,наляво и надясно), като всяко от движенията се задава с въвеждане на определен знак (натискане на определен клавиш). Трябва да се има предвид, че при движението на фигурата Ф някои от невидимите и части могат да станат видими и обратно, други да станат невидими.

**2 Задача /Национална олимпиада по информатика , Републикански кръг , март 1994 г. - сп."Обучението по математика и информатика" бр.3 1994 г. / за 8- тема 12 клас**

Разглежда се множеството P от точки с целочислени координати (x,y), за които $0<x<a$, $0<y<b$ и $a,b$ - цели числа.

Съставете програма, която решава следните задачи:

*Задача 1.* Да се въведат от текстов файл с име PTS.TXT данните за точките от множеството P. Файлът има следната структура:

| | | |
|---|---|---|
| 1-ред: | a | в |
| 2-ред: | $x_1$ | $y_1$ |
| ...... | | |
| последен (k+1)-ви ред | $x_k$ | $y_k$ |

Броят на точките от множеството P не е известен преди прочитане на данните от файла PTS.TXT.

Да се изобразят върху екрана съдържащите се в нея точки от множеството P.

*Задача 2.* Всяка от точките на множеството P да се оцвети в синьо или в червено по такъв начин, че разликата между сините и червените точки, разположени върху всяка права, успоредна на координатните оси, да не е по-голяма от 1. Върху екрана да се изобразят не повече от две оцветявания на точките от множеството P.

*Задача 3.*Да се построи (ако е възможно) само един правоъгълник D, който удовлетворява едновременно следните условия:

а)върховете му да са с целочислени координати и страните му да са успоредни на координатните оси;

б)броят на точките от множеството P, които са във вътрешноста на правоъгълника D да е равен на броя на точките, които са извън него.

Забележка:Точките, които принадлежат на страните на правоъгълника D не се считат нито вътрешни, нито външни.

**3 Задача / сп."Математика" бр.6 1968 г /**

В равнината са дадени права a и точка A, нележаща върху нея. Да се намери геометричното място на точка X в равнината, за която може да се намери такава точка У върху правата, че дължината на начупената линия XYA да е равна на дадена отсечка B.

**4 Задача /сп."Компютър" бр.12 1994 г. / от бр.7 1994 г Зад.4 от конкурса**

Даден е правоъгълник с целочислени дължини на страните a и в. Напишете програма, която проверява, дали е възможно той да бъде покрит (без застъпване и без излизане отвън) от n правоъгълника с целочислени дължини на страните $a_i$ и $b_i$, като всички разглеждани дължини на страни са различни помежду си.

**5 Задача /сп."Математика +" бр.1 1995 г./**

Една река пресича шосе на n места. Нито шосето, нито реката се самопресичат. Вървейки по шосето, номерираме пресечните точки последователно от 1 до n.Плувайки по реката, номерираме същите пресечни точки последователно пак с числата от 1 до n. Получава се следното съответствие: точката с номер 1 при първото номериране има номер P(1) при второто номериране, точката с номер 2 - P(2) и т.н. , точката с номер n - P(n). Съставете програма, която при зададено естествено число n :

1.Генерира всички възможни съответствия от описания вид.

2.При зададена пермутация P(1),P(2),...,P(n) на числата 1,2,. . .,n проверява дали това е съответствие от описания вид и изобразява графично как реката пресича шосето.

За колко голямо n вашата програма работи удовлетворително бързо?

**6 Задача /сп."Обучението по математика и информатика бр.4 1993 г. / тема "Мрежа от целочислени точки"**

Правоъгълникът ABCD е с върхове целочислени координати и страни, успоредни на координатните оси. Разглеждаме множеството N={-1,0,1}и множеството M от целочислени точки, които са вътрешни за ABCD или са върху страните му. На всяка точка M е съпоставено число от N.Да се състави програма, решаваща дадените по-долу задачи.

*Задача 1.*От първия ред на текстов файл се въвеждат координатите на върховете на правоъгълника ABCD. От останалите върхове се въвеждат числа от N , съответни на точки от M. Въвеждането се извършва по редове, като най-напред отляво-надясно се въвеждат числата, съответни на точките от страната

АВ, след това тези, съответни на точките от отсечката, успоредна на АВ, намираща се на разстояние 1 от нея и т.н., докато се достигне до CD.

*Задача 2.*Да се определи правоъгълник с максимална площ, чиито върхове са от М и на всеки от тях е съпоставено число 0.

*Задача 3.*Две точки от М ще наричаме съседни, ако са на разстояние 1 една от друга. Път между две точки Е и F от М е последователността от съседни точки на М, която започва с Е и завършва с F. Дължината на един път се определя като сума от числата, съответни на точките от пътя. Да се определи и изведе по подходящ начин върху екрана най-късият път (един от тях) между две дадени точки, чиито координати се въвеждат от клавиатурата.

*Задача 4.*Разглеждаме множеството от пет команди на компютърната костенурка:

-Orientation $x_i,y_i$ (ориентиране на костенурката с лице към точка $(x_i,y_i)$ - първоначална команда);

-MoveTo $x_i,y_i$ (придвижване на костенурката до съседната точка с координати $(x_i,y_i)$);

-TurnLeft (завъртане наляво на 90 градуса);

-TurnRight (завъртане надясно на 90 градуса );

Stop (преустановяване на движението - последна команда).

С така описаните команди да се генерира програма за придвижване на костенурката от точка Е в точка F, определени в задача 3.

*Задача 5* Да се изведе в текстов файл програмата, получена в задача 4 като всеки ред на файла съдържа една команда от програмата.

**7 задача /сп."Обучението по математика и информатика" бр.1 1993 г. / тема "Начупени линии"**

Дадено е множество от n ($n \geq 2$) точки в равнината с целочислени координати.

Две точки ще се считат за съседни, ако имат или равни абсциси, или равни ординати и между тях няма други точки от Р. Нека L е множеството на крайните списъци от точки $A_1,A_2,\ldots,A_i,A_{i+1},\ldots,A_m$, които съдържат всичките точки от Р и точките $A_i$ и $A_{i+1}$ са съседни ,където $i=1,2,\ldots,m-1$, $m \geq n$.

*Задача 1.*Съставете програма, която проверява дали множеството L е празното множество.

*Задача 2.*Съставете програма, която намира онези списъци от L, в които всяка точка от Р участвува само по един път.

*Задача 3.*Съставете програма, която намира онези списъци от L, които са върхове на начупена линия с минимална дължина.

*Задача 4.*Съставете програма, която намира онези списъци от L, които са върхове на начупена линия с минимална дължина и всяка точка от Р се среща само по един път.

*Задача 5.*Съставете програма, която намира онези списъци от L, в които всяка точка от Р участвува само по един път и съответните начупени линии, определени от тези точки са несамопресичащи се.

**8 Задача.**

Трасето за състезания е във вид на n-ъгълник , $n \geq 3$, в един от върховете на който се намира мястото за старт, а една от страните е линията на финала (мястото за старт не е на линията на финала). Пътят по трасето представлява начупена линия в n-ъгълника от старта към финала. Всеки отрязък на начупената линия се преминава за единица време и се явявя вектор на скоростта в този момент. В съседните моменти от време компонентите на вектора на скоростта са целочислени и или съвпадат или се различават с едно. Дължината на вектора на началната скорост е 0. Да се намери минималното време за преминаване през трасето. Да се намери минималния по дължина път по трасето.

**9 Задача. / Национален турнир по информатика , Пловдив май 1986 г. / тема за 10-11 клас**

*Задача 1.*Съставете програма, която въвежда координатите на 4 точки A,E,C,D (всички в първи квадрант), извършва обръщение към подпрограмите, описани в задачи 2,3,4 на тази тема и извежда получените от тях резултати.

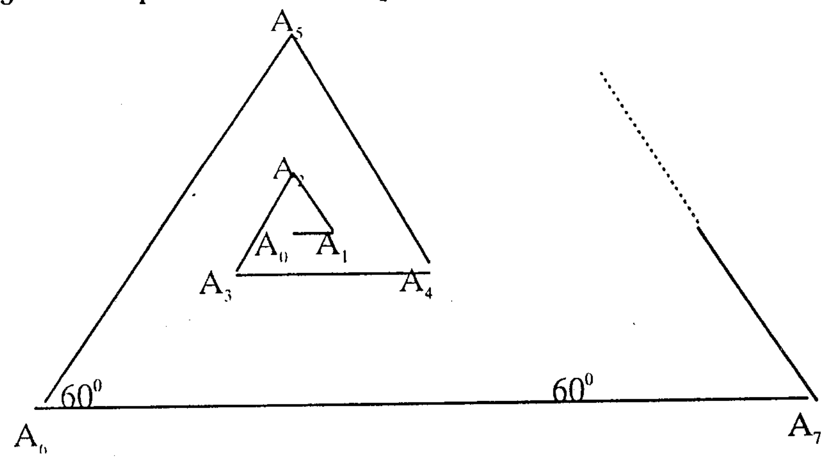
*Задача 2.*Съставете подпрограма,която проверява дали даден четириъгълник,определен с координатите на върховете си е квадрат.

*Задача 3.*Съставете подпрограма, която изчертава квадрат K, определен с координатите на върховете си, въвежда цялата положителна стойност D и изчертава затворена крива Ф, изцяло съдържаща K и имаща свойството, че всяка точка на K е на едно и също разстояние D до кривата Ф. Под разстояние на точката X от K до кривата Ф трябва да се разбира разстоянието между X и онази точка У от Ф , за която отсечката XУ е с възможно най-малка дължина.

*Задача 4.*Съставете подпрограма, която пресмята периметъра на кривата Ф и лицето на "коридора" , определен от K и Ф, т.е. пресмята лицето на частта от равнината, съдържаща се в кривата Ф и несъдържаща се в квадрата K.

**10 задача. /сп."Компютър" бр.6 1987 г. /**

Рагледайте триъгълната спирала, изобразена на фигурата.




Началото и е в точката $A_0(x_0,y_0)$, а дължината на първата отсечка $A_0A_1$ е h.Между дължините на първата и втората отсечка е в сила зависимостта: $A_1A2=k.A_0A_1$ $(1< k \le 2)$. Тази зависимост е валидна за кои да са две съседни страни (страни с обща точка) , т.е. $A_iA_{i+1}=k.A_{i-1}\ A_i$ $(i=1,2,\ldots)$.Ъгълът между всеки две съседни страни е 60 градуса.

1.Да се въведат стойности за $x_0$ ,$y_0$, h, k и на координатите на още една точка $P(x_p,y_p)$, като се извършва контрол за тяхната коректност.

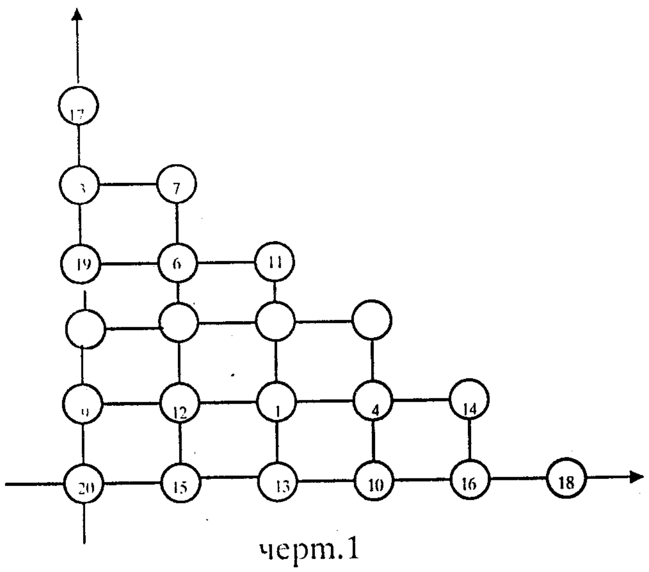
2.Да се начертае по възможност най-голяма част от спиралата, определена от параметрите си $(x_0,y_0)$ ,h и k.

3.Да се пресметне дължината на начертаната спирала (като сума от съставящите я отсечки) и да се определи триъгълника с върхове, явяващи се върхове на спиралата, чиито център на тежестта (пресечната точка на медианите) е най-отдалечен от P. (Една точка е връх на спиралата, ако е обща за две от страните и.)

## 11 Задача. / "Знаме на мира" Варна 1985 г. сп."Математика" бр. 5 1985 г.

В равнината е фиксирана правоъгълна координатна система и са дадени 21 точки, чиито координати (x,y) са цели числа, удовлетворяващи неравенствата $x \geq 0$, $y \geq 0$, $x+y \leq 5$. Във всяка от тези точки е поставена по една топка. Топките са номерирани с цели числа от 1 до 21 и всички са с различни номера. Да се състави програма, която:

а) въвежда координатите на точките и номерата на поставените върху тях топки, чисто първоначално разположение е дадено на черт.1;



черт.1

б) отпечатва координатите и номерата на топките, разположени както на черт.1;

в) намира и отпечатва координатите на и номерата на редицата от топки, започващи от топката с координати (0,0) ; за следващ член на редицата се избира топка с възможно най-малък номер, която да не е срещана преди това и да е на разстояние не по-голямо от квадратен корен от 2. (Разстоянието между две топки е разстоянието между точките, в които са поставени тези топки.)

г) преномерира топките по такъв начин, че определената в а) редица да съдържа всичките 21 топки.

## 3.Примерен тематичен план за школа по информатика

| | | |
|---|---|---|
| 1. | Теоритични сведения. | |
| 1.1. | Основни понятия. Геометричен елемент. Уравнение на права в равнината. Взаимно положение на две прави. Ъгъл между две прави. | **3 часа** |
| 1.2. | Метричен елемент. Пресмятане дължина на отсечка. Разстояние от точка до права. Пресмятане лице на триъгълник по зададени координати на върховете му. Пресмятане лицето на прост многоъгълник. | **3 часа** |
| 1.3. | Комбинаторен елемент. | |
| 1.3.1. | Пермутации. Лексикографско пораждане на пермутации без повторение. | **6 часа** |
| 1.3.2. | Вариации. Комбинации без повторение. Лексикографско пораждане на комбинации без повторение. | **6 часа** |
| 1.3.3. | Реализация на N вложени цикъла. | **6 часа** |
| 1.4. | Изготвяне на необходимите подпрограми и обединяването им в модул. | **3 часа** |
| 2. | Задачи от изпъкналост на геометрични фигури. | **6 часа** |
| 2.1. | Метрични задачи. Задачи, свързани с намиране на дължина на отсечка и ъгъл между две прави. | **6 часа** |
| 2.2. | Задачи за намиране лица и периметри на многоъгълници. | **6 часа** |
| 2.3. | Общи задачи. | |
| 2.3.1. | Задачи с прави. | **3 часа** |
| 2.3.2. | Задачи за окръжности. | **3 часа** |
| 2.3.3. | Задачи за многоъгълници. Триъгълници. | **6 часа** |
| 2.3.4. | Четириъгълници. | **6 часа** |
| 2.3.5. | Многоъгълници. Вътрешна точка за многоъгълник. | **6 часа** |
| 2.3.6. | Задачи с различни геометрични фигури. | **6 часа** |
| 2.3.7. | Пространствени задачи. Център на тежестта. | **6 часа** |
| 2.3.8. | Множества от точки. Операции с множества. | **6 часа** |
| 3. | Конкурсни задачи. | **3 часа** |

# ЛИТЕРАТУРА

1. Бърнев, П. ,Азълов, П. "Алгоритми", София, "Народна просвета" , 1978 г.

2. Димовски, И. "Ойлерови триангулации на изпъкнал многоъгълник" Сп. "Математика", бр. 3, 1988 9-13

3. Касьянов, В. Н. , Сабельфельд, В. К. "Сборник заданий по практикуму на ЭВМ" , Москва, Наука, 1986 г.

4. Липский, В. "Комбинаторика для програмистов" , Москва, "Мир" , 1988 г.

5. Милков, Д. "Алгоритми на някои геометрични задачи" сп. "Математика" бр. 5 1985 г. 13-18

6. Сидоренко, С. М. "Вычислительная геометрия в машиностроении" , Москва- "Машиностроение", 1983 г.

7. Целков, В. , Паргов, Д. "Алгоритми за изпъкнали многоъгълници" сп. "Математика" бр. 4 1986 г. 49-53

8. Чуканов, В. "Комбинаторика" , София, "Народна просвета" , 1977 г.