SZÁMÍTÓGÉPI GRAFIKA
VÁGÁS

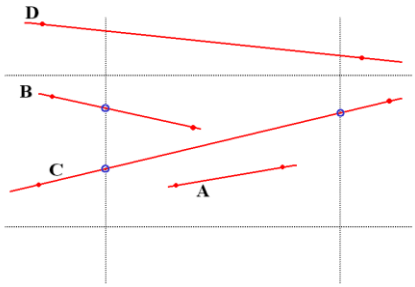
FELADAT:

Ha az alakzat nagyobb, mint a képtartomány♦, amelyben megjelenítendő, akkor a kívül eső részeket el kell hagyni, azaz az alakzatról „le kell vágni”, röviden szólva: az alakzatot *vág*ni kell. Tegyük föl, hogy az alakzat (egyenes) *szakas*zokra bontható. (A tartományok vágása hasonlóan megoldható, bár további problémákat is fölvet.) Az alábbiakban szakaszok vágását vizsgáljuk.

Cohen-Sutherland módszere

Cohen-Sutherland algoritmusa a következő felismerésre épít:
A szakasz és a képtartomány kölcsönös viszonya lényegében a következő négyféle lehet: a szakasz

1. teljesen benne van a képtartományban (*l. az ábrán az A szakaszt*),
2. csak egyik végpontja esik a tartományba (*l. az ábrán az B szakaszt*),
3. bár mindkét végpontja kívül esik a tartományon mégis a „középső” darabja a tartományban fut (*l. az ábrán az C szakaszt*),
4. és a tartomány nem rendelkezik közös ponttal (*l. az ábrán az D szakaszt*).



1. ábra. Szakasz és képtartomány viszonya.

Mindenek előtt próbálja ki: mire lehet gondolni (csvagasFP.exe, /CSvagasTP.exe)!

♦ Szokás szerint a képernyővel megegyező állású téglalapra gondolunk ***képtartomány*** esetén.

A lényeg

Osszuk fel a képtartományt magába foglaló síkot 9 részre, és sorszámozzuk meg az alábbiak szerint. Vegyük észre a sorszámokban a „bináris szabályszerűséget”!

$9=1001_2$	$1=0001_2$	$3=0011_2$
$8=1000_2$	$0=0000_2$	$2=0010_2$
$12=1100_2$	$4=0100_2$	$6=0110_2$

(-: Ugye megvan a bináris logika? :-)

Egy-egy bit tartozik –a felső, a „balsó”, az alsó és a „jobbsó”– a négy kívül levő síkrészhez.♠

A vágandó szakasz végpontjait kódoljuk a szerint, melyik tartományba esik. Ha a szakasz mindkét végpontja 0-kódú, akkor kirajzolható. Ha nem, akkor gondolkodni kell! :-)

Vegyük észre: akkor és csak akkor fut a szakasz „egyértelműen szerencsétlen” tartomány(ok)ba, ha a két végpont-kód *bites ÉS-művelet* eredménye *nem* 0 értékű. (Például: 9 ÉS 3 = 1001_2 ÉS 0011_2 = 0001_2… éppen az a bit 1-es, amelyik a „szerencsétlen” tartományhoz van rendelve.)

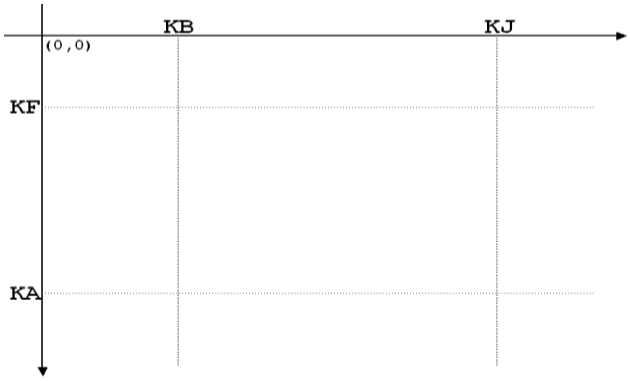
Ha egyéb eset áll fenn, akkor egyszer vagy kétszer metszi a (0-kódú) képtartomány határoló egyeneseit. A feladat e metszéspontok megtalálása. Valahogy így♥:

```
Eljárás CSVagas(Változó p1,p2:TPont):
    Változó
      c,c1,c2:TKod;  p:TPont
    c1:=VégpontKód(p1); c2:=VégpontKód(p2)
    Ciklus amíg nem ((c1=0) és (c2=0)) [legalább egyik végpont még nincs belül]
                 és ((c1 ÉS c2)=0)  [a szakasz nem biztos, hogy kívül fut]
      Ha c1=0 akkor [p1 OK. Nézzük a másikat: Csere(p1,p2)]
        p:=p1; p1:=p2; p2:=p;  c:=c1; c1:=c2; c2:=c
      Elágazás vége
      [p1 felöli új metszéspont meghatározása => p1]
      Elágazás
         (c1 ÉS 1)=1 esetén FelsoVagas(p1.x,p1.y,p2.x,p2.y)
         (c1 ÉS 2)=2 esetén JobbVagas(p1.x,p1.y,p2.x,p2.y)
         (c1 ÉS 4)=4 esetén AlsoVagas(p1.x,p1.y,p2.x,p2.y)
         (c1 ÉS 8)=8 esetén BalVagas(p1.x,p1.y,p2.x,p2.y)
      Elágazás vége
      c1:=VégpontKód(p1)
    Ciklus vége
```

♠ balsó ⇒ $1xxx_2$, alsó ⇒ $x1xx_2$, jobbsó ⇒ $xx1x_2$, felső ⇒ $xxx1_2$

♥ A logikai műveletek jelei kisbetűsek, a bit-műveletek NAGYBETŰSEK.

```
    Ha (c1=0) és (c2=0) akkor {a belső szakasz kirajzolható}
      Szakasz(p1,p2)
    Elágazás vége
Eljárás vége.
```



2. ábra. Képernyő-koordinátarendszer és képtartomány

```
Függvény VégpontKód(Konstans p:TPont):TKód
  Változó
    c:TKod
  c:=0
  Ha p.y<KF akkor c:=c VAGY 1
  Ha p.x>KJ akkor c:=c VAGY 2
  Ha p.y>KA akkor c:=c VAGY 4
  Ha p.x<KB akkor c:=c VAGY 8
  VégpontKód:=c
Eljárás vége.
```

Az egyes vágáseljárások meghatározzák a `p1` és `p2` pontra illeszkedő egyenes és a megfelelő határoló egyenes metszéspontját. Ez két lineáris egyenletből álló egyenletrendszer megoldását jelenti.

Például a `FelsőVágás` esetében: az $y=(y_2-y_1)/(x_2-x_1)*(x-x_1)+y_1$ és az $y=KF$ egyenletek megoldását keressük. Azaz x-re kell rendezni a $KF=(y_2-y_1)/(x_2-x_1)*(x-x_1)+y_1$ egyenletet, amiből már következik a keresett x és y:

$$x=x_1+(x_2-x_1)/(y_2-y_1)*(KF-y_1)$$

$$y=KF$$

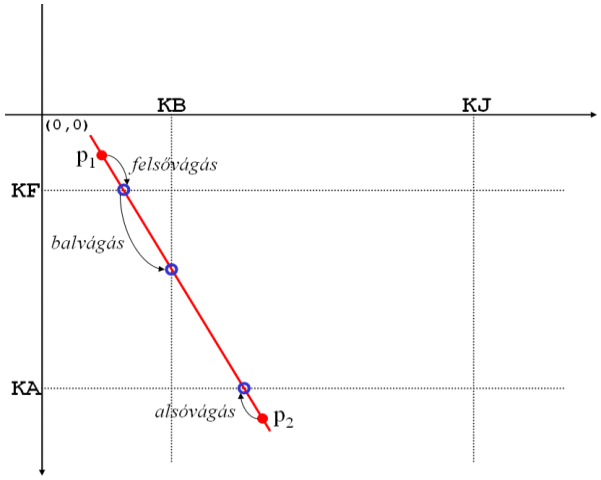
Azaz az eljárás kódja:

```
Eljárás FelsoVagas(Változó x1,y1,x2,y2:Egész[1]):
  x1:=x1+(x2-x1)/(y2-y1)*(KF-y1); y1:=KF
Eljárás vége.
```

A program az alábbi deklarációk után válik teljesen érthetővé, egyértelművé:

```
Típus  TPont=Rekord(x,y:Egész)
       TKod=0..12
```

[1] Világos, hogy valójában csak az 1. pont koordinátái fognak változni, így precízebb lenne az alábbi szignatúra:
`Eljárás FeloVagas(Változó x1,y1:Egész, Konstans x2,y2:Egész)`



3. ábra. A Cohen-Sutherland algoritmusa működés közben.

A (Turbo/Free Pascal) keretprogram olvasható alább, amelyben csak néhány eljárás törzse maradt kitöltendő. (Letölthető a Free Pascal-hoz: LetoltendoVagasFP.zip)

A KERETPROGRAM

```
1   Program Vagas;
2     Uses
3       Crt,Graph;
4     Const
5       gPath='C:\langs\tp\bgi';
6       {KépTartományt határoló egyenesek paraméterei (VGA-hoz igazítva):}
7       KF0=100; KB0=100; KJ0=500; KA0=350;
8       MaxX0=640; MaxY0=480;      {Próbaszakaszok száma:}
9       SzakaszDb=5;
10    Type
11      TPont=Record x,y:Integer End;
12      TSzakasz=Array [1..2] of TPont;
13      TKod=0..12;
14    Const {Próbaszakaszok:}
15      sz:Array [1..SzakaszDb] of TSzakasz=(
16                                  ((x:KB0+229;y:KF0+129),(x:KJ0-229;y:KA0-129)),
17                                  ((x:KB0-19;y:KF0+19),(x:KB0+190;y:KF0+39)),
18                                  ((x:KB0-9;y:KA0-19),(x:KJ0+19;y:KA0-99)),
19                                  ((x:KB0-9;y:KF0-9),(x:KJ0+19;y:KF0-19)),
20                                  ((x:KB0-59;y:KA0+59),(x:KJ0+69;y:KF0-69))
21                                 );
22    Var
23      p1,p2:TPont;
24      i:Integer;
25      KF,KB,KA,KJ:Integer;
26
```

```
27        Procedure Inic;
28          Var
29            i,j,
30            gd,gm:Integer;
31            nyX,nyY:Real;
32        Begin
33          DetectGraph(gd,gm);
34          InitGraph(gd,gm,gPath);
35          ClearDevice;
36          {képernyőre transzformálás (TP->FP miatt):}
37          nyX:=GetMaxX/MaxX0; nyY:=GetMaxY/MaxY0;
38          KF:=Round(KF0*nyY); KA:=Round(KA0*nyY);
39          KB:=Round(KB0*nyX); KJ:=Round(KJ0*nyX);
40          For i:=1 to SzakaszDb do
41            For j:=1 to 2 do
42            Begin
43              sz[i][j].x:=Round(sz[i][j].x*nyX); sz[i][j].y:=Round(sz[i][j].y*nyY);
44            End;
45          SetLineStyle(SolidLn, 0, ThickWidth);
46          SetColor(Green); Rectangle(KB,KF,KJ,KA);
47        End;
48
49        Procedure Szakasz(Const p1,p2:TPont);
50        Begin
51          Line(p1.x,p1.y,p2.x,p2.y);
52        End;
53
54        Function VegpontKod(Const p:TPont):TKod;
55        Begin
56
57        End;
58
59        Procedure FelsoVagas(Var x1,y1,x2,y2:Integer);
60        Begin
61
62        End;
63
64        Procedure AlsoVagas(Var x1,y1,x2,y2:Integer);
65        Begin
66
67        End;
68
69        Procedure BalVagas(Var x1,y1,x2,y2:Integer);
70        Begin
71
72        End;
73
74        Procedure JobbVagas(Var x1,y1,x2,y2:Integer);
75        Begin
76
77        End;
```

```
78
79        Procedure CSVagas(Var p1,p2:TPont);
80          Var
81            c,c1,c2:TKod;
82            p:TPont;
83        Begin
84
85        End;
86
87      Begin
88        Inic;
89        For i:=1 to SzakaszDb do
90        Begin
91          SetColor(Red); p1:=sz[i][1]; p2:=sz[i][2];
92          Szakasz(p1,p2);
93          ReadKey;
94          CSVagas(p1,p2);
95          ReadKey
96        End;
97      End.
```

Minden egyben letölthető: Vagas.zip.

A MEGOLDÁS:

```
1   Program Vagas;
2     Uses
3       Crt,Graph;
4     Const
5       gPath='C:\langs\tp\bgi';
6       {KépTartományt határoló egyenesek paraméterei (VGA-hoz igazítva):}
7       KF0=100; KB0=100; KJ0=500; KA0=350;
8       MaxX0=640; MaxY0=480;
9       {Próbaszakaszok száma:}
10      SzakaszDb=5;
11    Type
12      TPont=Record x,y:Integer End;
13      TSzakasz=Array [1..2] of TPont;
14      TKod=0..12;
15    Const
16      sz:Array [1..SzakaszDb] of TSzakasz=(
17                                  ((x:KB0+229;y:KF0+129),(x:KJ0-229;y:KA0-129)),
18                                  ((x:KB0-19;y:KF0+19),(x:KB0+190;y:KF0+39)),
19                                  ((x:KB0-9;y:KA0-19),(x:KJ0+19;y:KA0-99)),
20                                  ((x:KB0-9;y:KF0-9),(x:KJ0+19;y:KF0-19)),
21                                  ((x:KB0-59;y:KA0+59),(x:KJ0+69;y:KF0-69))
22                                  );
23    Var
24      p1,p2:TPont;
25      i:Integer;
26      KF,KB,KA,KJ:Integer;
27
28    Procedure Inic;
29      Var
30        i,j,
31        gd,gm:Integer;
32        nyX,nyY:Real;
33    Begin
34      DetectGraph(gd,gm);
35      InitGraph(gd,gm,gPath);
36      ClearDevice;
37      {képernyőre transzformálás (TP->FP miatt):}
38      nyX:=GetMaxX/MaxX0; nyY:=GetMaxY/MaxY0;
39      KF:=Round(KF0*nyY); KA:=Round(KA0*nyY);
40      KB:=Round(KB0*nyX); KJ:=Round(KJ0*nyX);
41      For i:=1 to SzakaszDb do
42        For j:=1 to 2 do
43        Begin
44          sz[i,j].x:=Round(sz[i,j].x*nyX); sz[i,j].y:=Round(sz[i,j].y*nyY);
45        End;
```

```
46          SetLineStyle(SolidLn, 0, ThickWidth);
47          SetColor(Green); Rectangle(KB,KF,KJ,KA);
48        End;
49
50        Procedure Szakasz(Const p1,p2:TPont);
51        Begin
52          Line(p1.x,p1.y,p2.x,p2.y);
53        End;
54
55        Function VegpontKod(Const p:TPont):TKod;
56        Var
57          c:TKod;
58        Begin
59          c:=0;
60          If p.y<KF then c:=c OR 1;
61          If p.x>KJ then c:=c OR 2;
62          If p.y>KA then c:=c OR 4;
63          If p.x<KB then c:=c OR 8;
64          VegpontKod:=c
65        End;
66
67        Procedure FelsoVagas(Var x1,y1,x2,y2:Integer);
68        Begin
69          x1:=Round(x1+(x2-x1)/(y2-y1)*(KF-y1)); ♣
70          y1:=KF
71        End;
72
73        Procedure AlsoVagas(Var x1,y1,x2,y2:Integer);
74        Begin
75          x1:=Round(x1+(x2-x1)/(y2-y1)*(KA-y1));
76          y1:=KA
77        End;
78
79        Procedure BalVagas(Var x1,y1,x2,y2:Integer);
80        Begin
81          y1:=Round(y1+(y2-y1)/(x2-x1)*(KB-x1));
82          x1:=KB
83        End;
84
85        Procedure JobbVagas(Var x1,y1,x2,y2:Integer);
86        Begin
87          y1:=Round(y1+(y2-y1)/(x2-x1)*(KJ-x1));
88          x1:=KJ
89        End;
90
91        Procedure CSVagas(Var p1,p2:TPont);
92          Var
93            c,c1,c2:TKod;
94            p:TPont;
```

♣ Hogy ne legyen a részletszámítások során sem túlcsordulás az első adandó helyen osztunk.

```
95        Begin
96          VegpontKod(c1,p1); VegpontKod(c2,p2);
97          While not ((c1=0) and (c2=0)) {legalább egyik még nincs belül}
98                and ((c1 AND c2)=0)  {nem biztos, hogy kívül} do
99          Begin
100           If c1=0 then {p1 OK. Nézzük a másikat: Csere(p1,p2)}
101           Begin
102             c:=c1; c1:=c2; c2:=c;
103             p:=p1; p1:=p2; p2:=p;
104           End;
105           {p1 felöli új metszéspont meghatározása => p1}
106           If
107             (c1 AND 1)=1 then FelsoVagas(p1.x,p1.y,p2.x,p2.y) else if
108             (c1 AND 2)=2 then JobbVagas(p1.x,p1.y,p2.x,p2.y) else if
109             (c1 AND 4)=4 then AlsoVagas(p1.x,p1.y,p2.x,p2.y) else if
110             (c1 AND 8)=8 then BalVagas(p1.x,p1.y,p2.x,p2.y)
111           {EndIf};
112           VegpontKod(c1,p1);
113           {Metszéspont-keresés -- nyomkövetés:}
114           SetLineStyle(CenterLn, 0, NormWidth);
115           SetColor(Yellow);
116           Szakasz(p1,p2);
117           ReadKey;
118           SetLineStyle(SolidLn, 0, ThickWidth);
119           {Nyomkövetés vége}
120         End;
121         If (c1=0) and (c2=0) then
122         Begin {a belső szakasz kirajzolható}
123           SetColor(Blue);
124           Szakasz(p1,p2);
125         End;
126       End;
127
128   Begin
129     Inic;
130     For i:=1 to SzakaszDb do
131     Begin
132       SetColor(Red); p1:=sz[i][1]; p2:=sz[i][2];
133       Szakasz(p1,p2);
134       ReadKey;
135       CSVagas(p1,p2);
136       ReadKey
137     End;
138   End.
```