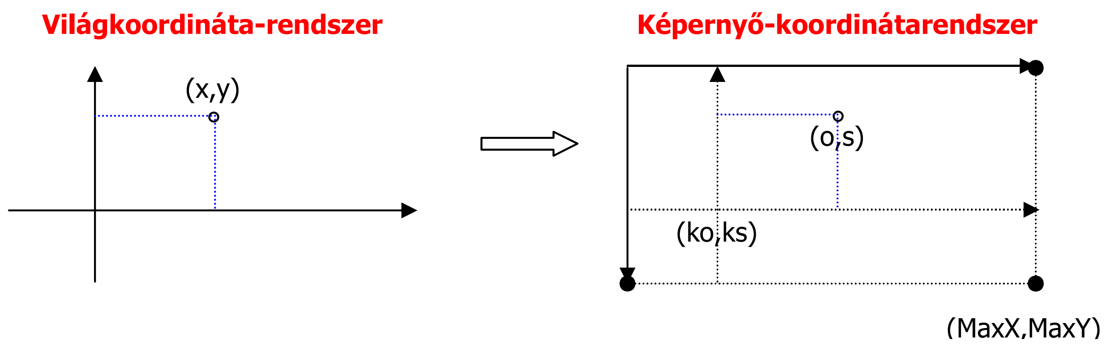
GÖRBEÍV-RAJZOLÁS

TARTALOM

0 KIINDULÓPONT

A képernyőn a „normál” koordinátarendszer:

- Origó a bal-felső sarokban.
- A pixel az egység.
- A görbék, ívek csak egész koordinátájú pontjaival foglalkozunk.



A két koordinátarendszer közötti áttérést ráhagyjuk a `PontRajzol` eljárásra, amely felhasználja a képernyőre pontot rajzoló `Pont` utasítást [1]:

```
Eljárás PontRajzolás(Konstans x,y:Valós):
  [globális input: ks, ko]
  s:=Kerekít(ks-y); o:=Kerekít(ko+x) [2]
  Ha s∈[0,MaxY] és o∈[0,MaxX] akkor Pont(o,s)
Eljárás vége.
```

1 SZAKASZRAJZOLÁS

A feladat: szakaszt rajzolni (x_1,y_1) és (x_2,y_2) között. Föltehető, hogy $x_1 \le x_2$.

1.1 Naiv megoldás

A két ponton húzható egyenes egyenlete: $y=(y_2-y_1)/(x_2-x_1)*(x-x_1)+y_1$

A megoldás lényege:

1. vegyük sorra x lehetséges (egész) értékeit $[x_1,x_2]$ között, és
2. rajzoljuk ki az $(x,y(x))$ pontot!

[1] ami a TurboGrafikában (Graph unit a Turbo/FreePascal-ban): `PutPixel(o,s,pontSzín)`.

[2] Érdemes eljátszadozni a `Kerekít` függvény helyett `Egészrész`-szel.

```
Eljárás SzakaszRajzolás(Konstans x1,y1,x2,y2:Egész):
  Változó
    x,y,it:Valós
  it:=(y2-y1)/(x2-x1)   [iránytangens]
  Ciklus x=x1-től x2-ig [3]
    y:=(x-x1)*it+y1; PontRajzolás(x,y)
  Ciklus vége
Eljárás vége.
```

Problémák:

1. $x_1=x_2$–re nem működik.
2. $it\le 1$ (legfeljebb 45° lejtésszög) esetén „folytonos” pixelek sorozata a szakasz, $it>1$ (több, mint 45° lejtésszög) esetén „szakadozott” pixelek sorozata.
3. a lejtéssel változó fényerő:

   Pl.: it=0, akkor a „fénysűrűség”=(N-fénypont)/(N pixelnyi hossz)=1,

   it=1, akkor a „fénysűrűség”=(N-fénypont)/($\sqrt{2}$ *(N pixelnyi hossz))= $\sqrt{2}/2\approx 0{,}71$.

Az alábbiakban először az első két problémára keresünk megoldást.

1.2 Első javítás

Válasszuk szét az `it≤1`, `it>1`, `és` $x_1=x_2$ eseteket! A 2. esetben cseréljük föl `x` és `y` szerepét, a 3. esetben speciálisan erre az esetre specifikálunk egy eljárást.

```
Eljárás SzakaszRajzolás(Konstans x1,y1,x2,y2:Egész):
  Változó
    x,y,it:Valós
  Ha x1=x2 akkor
    FüggőlegesRajzolás(x1,y1,y2)
   különben
    it:=(y2-y1)/(x2-x1)   [iránytangens]
    Ha it≤1 akkor
        Ciklus x=x1-től x2-ig
          y:=(x-x1)*it+y1; PontRajzolás(x,y)
        Ciklus vége
      különben
        Ciklus y=y1-től y2-ig
          x:=(y-y1)/it+x1; PontRajzolás(x,y)
        Ciklus vége
    Elágazás vége
  Elágazás vége
Eljárás vége.
```

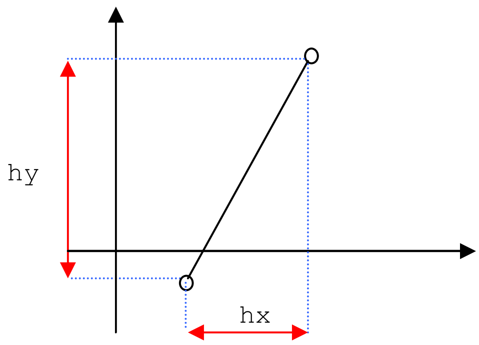
1.3 Második javítás

Válasszuk meg úgy az x-irányú lépésközt, hogy az megfelelő legyen minden esetben (a függőleges esetet kivéve). Ez elérhető, ha az x-irányú eltérés (`hx`) és az y-irányú eltérés (`hy`) maximumával normáljuk a lépésközöket.

---

[3] Ha $x_1\le x_2$.nem lenne feltehető, akkor a ciklus `Sgn(x2-x1)`-esével szervezzük

```
h:=Max(|hx|,|hy|)
lx:=hx/h     – x-lépésköz
ly:=hy/h     – y-lépésköz
  ⇒
lx,ly≤1
  ⇒
Max(lx,ly)=1
```



Hány lépést (k) kell tenni?

$x_2=x_1+k*lx \Leftrightarrow k=(x_2-x_1)/lx \Leftrightarrow k=(x_2-x_1)/(hx/h) \Leftrightarrow$

$\Leftrightarrow k=(x_2-x_1)*h/hx \Leftrightarrow k=(x_2-x_1)*h/(x_2-x_1) \Leftrightarrow k=h$

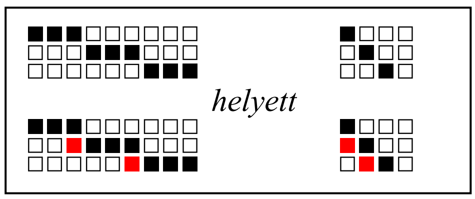
Hasonlóan (és nem meglepő módon) ugyanezt kapjuk az y-irányban is.

```
Eljárás SzakaszRajzolás(Konstans x1,y1,x2,y2:Egész):
  Változó
    lx,ly,x,y:Valós
    k,hx,hy,h:Egész
  hx:=x2-x1; hy:=y2-y1
  Ha Abs(hx)>Abs(hy) akkor h:=Abs(hx) különben h:=Abs(hy)
  Ha h=0 akkor
    PontRajzolás(x1,y1)
   különben
    lx:=hx/h   [x-irányú lépésköz]
    ly:=hy/h   [y-irányú lépésköz]
    x:=x1; y:=y1; PontRajzolás(x1,y1)   [kezdőpont]
    Ciklus k=1-től h-ig
      x:+lx; y:+ly; PontRajzolás(x,y)
    Ciklus vége
  Elágazás vége
Eljárás vége.
```

1.4 Javítás a fényerő-problémán

Ötlet:

kiegészíteni *további* (azonos színű, fényességű) pontokkal a „töréseknél". (A színezés csak az új pontok kiemelését szolgálja.)



Ami után a probléma „megfordul":

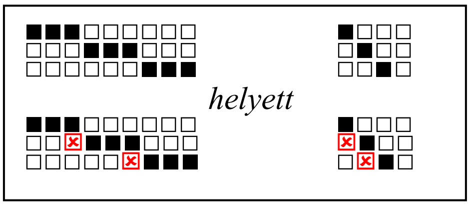
Ui.. ha it=1, akkor h pont helyett h+(h-1) pont lesz,

így a fénysűrűség= ((2*h-1)-fénypont)/($\sqrt{2}$ *(h pixelnyi hossz))≈

≈ ((2*h)-fénypont)/($\sqrt{2}$ *(h pixelnyi hossz))= $\sqrt{2}$ *(h-fénypont)/(h pixelnyi hossz)= $\sqrt{2}$ ≈1,41

Ötlet:

kiegészíteni *kisebb* (számítható) fényerejű pontokkal.



2 KÖRRAJZOLÁS

A feladat körívet rajzolni. Vegyük észre, hogy

1. a kör szimmetriája miatt, ha az `(x,y)` pont rajta van az íven , akkor az `(-x,y)`, `(x,-y)`, `(-x,-y)` `pontok` is rajta lesznek. (Sőt további szimmetriatengelyei is vannak, amelyek kihasználhatók!)
2. az (x_0, y_0) középpontú kör a `(0,0)` középpontú eltolásával egyszerűen megkapható, amelyet ismét rábízhatunk a `PontRajzol` eljárásra.

2.1 A körív pontjai: (x, $\sqrt{r^2 - x^2}$)

Mivel a körív pontjai kielégítik az $y^2=r^2-x^2$ egyenletet, kapjuk a kézen fekvő megoldást:

```
Eljárás KörRajzolás(Konstans r:Egész):
  Ciklus x=0-tól r-ig
    y:=Egész(Négyzetgyök(r*r-x*x))
    PontRajzolás(x,y); PontRajzolás(-x,y)
    PontRajzolás(x,-y); PontRajzolás(-x,-y)
  Ciklus vége
Eljárás vége.
```

Probléma:

Az ív „meredek” érintőjű részén szakadozik.

2.2 A körív pontjai: (r*cos(α), r*sin(α))

Ezen a problémán segíthetünk azzal, hogy „egyenletesen” járjuk be a negyed körívet. Nem x-szerint haladunk, hanem origóból azonos (Δα) szögelfordulással haladva:

```
Eljárás KörRajzolás(Konstans r:Egész):
  Változó
    Δα,α: Valós
  Δα:=???
  Ciklus α=0-tól π/2-ig Δα-asával
    x:=r*cos(α); y:=r*sin(α)
    PontRajzolás(x,y); PontRajzolás(-x,y)
    PontRajzolás(x,-y); PontRajzolás(-x,-y)
  Ciklus vége
Eljárás vége.
```

Meggondolandó a Δα választása! Ha túl kicsire választjuk, akkor sokszor fogja ugyanazt a pontot kiszínezni, ha meg túl nagyra, akkor meg kimaradnak pontok.

Legyen Δα-nyi fordulat az r-sugarú íven kb. 1 pixelnyi! Azaz **Δα / (2*π) = 1 / (2*r*π)**

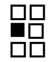
⇒ Δα = 1 / r

2.3 A körív követése

Egy, a görbék rajzolására általánosan alkalmazható ötlet:

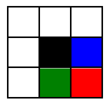
1. kiindulás a görbe egy alkalmas kezdőpontjából,
2. válasszunk valamilyen elképzelhető haladási (rajzolási) irányt (balra/jobbra és/vagy fel/le),
3. az irányba eső szomszédos (mondjuk: 5) pontokat vizsgáljuk meg: melyik tér el legkevésbé a görbétől [4]. S arra lépjünk tovább!

Pl.:



A kör esetén megint csak az első negyedbeli ívét használjuk föl „vezérlőként”.

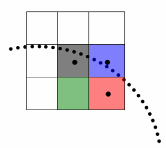
```
Eljárás KörRajzolás(Konstans r:Egész):
  Változó
    x,y,rr:Valós
  x:=0; y:=r; rr:=r*r
  Ciklus amíg y≥0
    PontRajzolás(x,y); PontRajzolás(-x,y)
    PontRajzolás(x,-y); PontRajzolás(-x,-y)
    Elágazás
      (x+1)*(x+1)+y*y≤rr esetén
            x:+1              [vízszintesen]
      (x+1)*(x+1)+(y-1)*(y-1)≤rr esetén
            y:-1; x:+1        [vízszintesen és le]
      egyéb esetben
            y:-1              [le]
    Elágazás vége
  Ciklus vége
Eljárás vége.
```



Megjegyzés:

Az világos, hogy az algoritmusbeli feltételek –sorrendjükkel együtt– feltételezik, hogy a „vezérlő szerepű” körív-darab éppen az ***1. negyedbe eső negyed körív***, sőt, hogy a rajzolás az `(0,r)`-től kezdődik. (Ui.: `y≥0`; és elegendő 5 helyett a rajzon jelzett 3 „próbapont”; a „*leg-kevésbé eltérés*”-t helyettesíthetjük a „*a körlapba először belépés*”-sel.)

Észre kell vennünk, hogy az általános algoritmust csak „ε-pontossággal” követi a fenti, ui. elképzelhető, hogy bár a jobbra lépés lenne a legjobb, mi mégis a „jobbra-le” szomszédot választjuk, mert ő az első még bennmaradó. Ez a hiba azonban a kör „körségét” nem befolyásolja, az ívtől való elszakadástól nem kell tartanunk.



[4] Ez egy jó, nem ritkán nehezen megválaszolható kérdés! Gyakran helyettesítik valami hasonló, de könnyebben megválaszolhatóval.