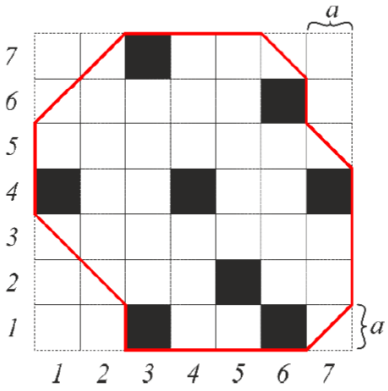
Zadatak: Janjad

70 bodova



Bojan najviše voli slikati janjetinu[1]. Kako bi u janjetini mogao uživati svaki dan, kupio je izvjesnu količinu žive janjadi. Bojan je primijetio da, ako podijeli svoje dvorište na mrežu kvadratnih zona, svako janje ima svoju omiljenu zonu kvadratnog oblika. Bojan želi ograditi dvorište (kako janjad ne bi pobjegla), ali tako da omiljene zone svakog janjca budu sadržane unutar ograde.

Napišite proceduru `JANJAD :l :a` koja crta kvadratne zone koje voli Bojanova janjad i crvenu (`"RED`) ogradu najmanjeg mogućeg opsega oko tih zona. Dvorište je podijeljeno na kvadrate (kao na skici), a omiljene zone su zadane listom `:l` koja se sastoji od podlisti koje sadržavaju dva elementa: redak i stupac kvadrata koji je omiljen nekom janjetu. Ograda se može graditi samo po stranicama ili dijagonalama kvadrata na koje je podijeljeno dvorište. U slučaju da postoji više ograda s istim minimalnim opsegom, potrebno je nacrtati bilo koju. Skica odgovara drugom primjeru test podataka.

Ulazni podaci

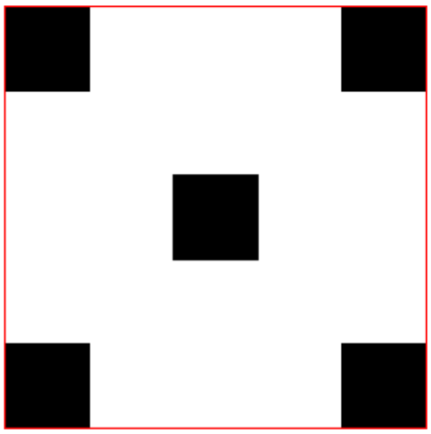
Lista `:l` je neprazna lista koja se sastoji od podlisti koje sadržavaju po dva prirodna broja. Varijabla `:a` je prirodan broj.

Bodovanje

U test podacima vrijednim 30% (21) bodova, ograda će imati oblik pravokutnika, omiljene zone nalazit će se u njegovim kutovima.

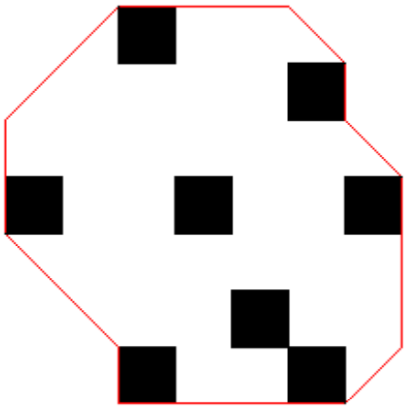
Primjeri test podataka

```
CS JANJAD [[1 1] [5 1] [5 5] [1 5] [3 3]] 50
```



```
CS JANJAD [[1 3] [2 5] [4 1] [4 4] [4 7] [7 3] [6 6] [1 6]] 30
```

CS JANJAD [[1 3] [2 5] [4 1] [4 4] [4 7] [7 3] [6 6] [1 6]] 30



CS JANJAD [[1 1] [5 1] [5 5] [1 5] [3 3]] 50

CS JANJAD [[1 3] [2 5] [4 1] [4 4] [4 7] [7 3] [6 6] [1 6]] 30

Crtanje kvadrata

```
to janjad :l :a
      nacrtaj_kvadrate

end

to kvadrat
      repeat 4 [fd :a rt 90]
      pu fd 3 rt 90 fd 3 pd
      fill
      pu bk 3 lt 90 bk 3 pd
end

to nacrtaj_kvadrate
      setpc 0
      make "poz pos
      foreach :l [
            pu rt 90 fd ((last ?)-1)*:a lt 90 fd ((first ?)-1)*:a pd
            kvadrat
            pu bk ((first ?)-1)*:a rt 90 bk ((last ?)-1)*:a lt 90 pd
      ]
End
```

Poziv procedure

```
CS slowdraw 50 JANJAD [[1 1] [5 1] [5 5] [1 5] [3 3]] 50
CS slowdraw 50 JANJAD [[1 3] [2 5] [4 1] [4 4] [4 7] [7 3] [6 6] [1 6]]
30
```

https://web.microsoftstream.com/video/8d4b6066-ca72-44e6-86c9-35fdbf0d9bea

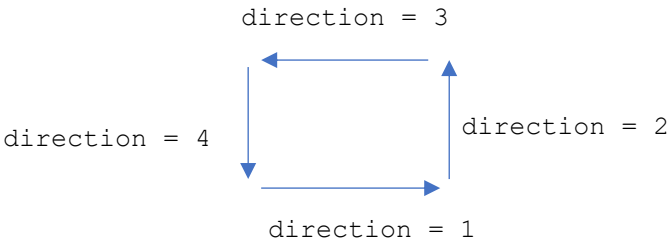
Drugi način (pomoću array):

```
to janjad :l :a
      make "r 0
      make "s 0
      foreach :l[
            if (first ?)>:r[make "r first ?]
            if (last ?)>:s[make "s last ?]
      ]
```

```
      nacrtaj_kvadrate :r :s :l

end

to kvadrat
      repeat 4 [fd :a rt 90]
      pu fd 3 rt 90 fd 3 pd
      fill
      pu bk 3 lt 90 bk 3 pd
end

to matrica :m :n :l
make "v list :n :m
make "myarray (MDARRAY :v 1)
for [j 1 :n][
      for [k 1 :m][
            make "o list :j :k
            mdsetitem :o :myarray 0
      ]
]
for [j 1 :n][
      for [k 1 :m][
            make "o list :j :k
            for [i 1 [count :l]][
                  make "c item :i :l
                  if (equalp :c :o)[mdsetitem :o :myarray 1]
            ]
      ]
]
end

to nacrtaj_kvadrate :r :s :l
      setpc 0
      matrica :r :s :l
      for [j 1 :r][
            make "p pos
            for [k 1 :s][
                  if (mditem (list :j :k) :myarray)=1[kvadrat]
                  pu rt 90 fd :a lt 90 ppt
            ]
            pu setpos :p
            pu fd :a ppt
      ]
End
```

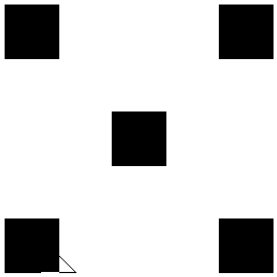
https://web.microsoftstream.com/video/7a6d1dc0-2dbb-4321-b24e-599c141e451c

Crtanje ograde



[1 1] [5 1] [5 5] [1 5] [3 3]

Pomak na najmanju poziciju



Crtanje crvene linije
https://web.microsoftstream.com/video/ae111f0d-ef6e-47b7-b457-64aaafa1a1da

https://web.microsoftstream.com/video/3449ebb6-086d-403f-96a8-9c38b7c10ee6

Kada bi ograda bila samo ravna onda bi ovo bilo rješenje:

```
;Traženje najmanjeg prema prvom clanu podliste (redu)
      make "min_r first :l
      foreach :l [
            if (first ?) < (first :min_r) [
                  make "min_r ?
            ]
      ]
      setpc "red
      ;Prvi pomak
      pu rt 90 fd :a*(last :min_r) lt 90 fd :a*((first :min_r)-1) pd
      make "r (first :min_r)-1
      make "s (last :min_r)
      make "direction 1
      while [:direction < 5] [
            ; provjeri postoji li u trenutnom redu/stupcu neki kvadrat
            foreach :l [
                  if :direction = 1 [
                        if (and (first ?) = :r+1 (last ?) > :s) [
                              rt 90 fd :a*((last ?)-:s) lt 90
                              make "s last ?
                        ]
                  ]
                  if :direction = 2 [
                        if (and (last ?) = :s (first ?) > :r) [
```

```
                        rt 90 fd :a*((first ?)-:r) lt 90
                        make "r first ?
                    ]
                ]
                if :direction = 3 [
                    if (and (first ?) = :r (last ?) <= :s) [
                        rt 90 fd :a*(:s-((last ?)-1)) lt 90
                        make "s (last ?)-1
                    ]
                ]
                if :direction = 4 [
                    if (and (last ?) = :s+1 (first ?) <= :r) [
                        rt 90 fd :a*(:r-((first ?)-1)) lt 90
                        make "r (first ?)-1
                    ]
                ]
            ]

            make "change_direction "true
            ifelse :change_direction [
                lt 90
                make "direction :direction+1
            ] [

            ]
        ]
        rt 90 fd :a*((last :min_r)-:s) lt 90
end
```

Međutim ograda se sastoji i od dijagonalnih linija. Pa je paralelno za svaki član potrebno provjeravati je li član >:s ili <:s te >:r ili <:r.

```
; crtanje dijagonalne crvene linije
            make "change_direction "true
            foreach :l [
                if :direction = 1 [
                    if (last ?) > :s [
                        make "change_direction "false
                        make "iduci ?
                    ]
                ]
                if :direction = 2 [
                    if (first ?) > :r [
                        make "change_direction "false
                        make "iduci ?
                    ]
                ]
                if :direction = 3 [
                    if (last ?) <= :s [
                        make "change_direction "false
                        make "iduci ?
                    ]
                ]
                if :direction = 4 [
                    if (first ?) <= :r [
                        make "change_direction "false
```

```
                        make "iduci ?
                    ]
                ]
            ]
```

Ako se to dogodi tada je change_direction="false pa će program ući u else slučaj i napraviti pomak.

```
ifelse :change_direction [
                lt 90
                make "direction :direction+1
            ] [
                ; pomak
                make "poz pos
                pu rt 90 fd :a lt 90 fd :a pd
                setpos :poz
                pu rt 90 fd :a lt 90 fd :a pd
                if :direction = 1 [
                    make "r :r+1 ;Dijagonala gore desno
                    make "s :s+1
                ]
                if :direction = 2 [
                    make "r :r+1 ;Dijagonala gore lijevo
                    make "s :s-1
                ]
                if :direction = 3 [
                    make "r :r-1 ;Dijagonala dolje lijevo
                    make "s :s-1
                ]
                if :direction = 4 [
                    make "r :r-1 ;Dijagonala dolje desno
                    make "s :s+1
                ]
            ]
```

Cijelo rješenje:

```
to janjad :l :a
     nacrtaj_kvadrate
     ;Traženje najmanjeg prema prvom clanu podliste (redu)
     make "min_r first :l
     foreach :l [
          if (first ?) < (first :min_r) [
               make "min_r ?
          ]
     ]
     setpc "red
;Prvi pomak
     pu rt 90 fd :a*(last :min_r) lt 90 fd :a*((first :min_r)-1) pd
     make "r (first :min_r)-1
     make "s (last :min_r)
     make "direction 1



     while [:direction < 5] [
          ; provjeri postoji li u trenutnom redu/stupcu neki kvadrat
```

```
foreach :l [
     if :direction = 1 [
          if (and (first ?) = :r+1 (last ?) > :s) [
               rt 90 fd :a*((last ?)-:s) lt 90
               make "s last ?
          ]
     ]
     if :direction = 2 [
          if (and (last ?) = :s (first ?) > :r) [
               rt 90 fd :a*((first ?)-:r) lt 90
               make "r first ?
          ]
     ]
     if :direction = 3 [
          if (and (first ?) = :r (last ?) <= :s) [
               rt 90 fd :a*(:s-((last ?)-1)) lt 90
               make "s (last ?)-1
          ]
     ]
     if :direction = 4 [
          if (and (last ?) = :s+1 (first ?) <= :r) [
               rt 90 fd :a*(:r-((first ?)-1)) lt 90
               make "r (first ?)-1
          ]
     ]
]
; crtanje dijagonalne crvene linije
make "change_direction "true
foreach :l [
     if :direction = 1 [
          if (last ?) > :s [
               make "change_direction "false
               make "iduci ?
          ]
     ]
     if :direction = 2 [
          if (first ?) > :r [
               make "change_direction "false
               make "iduci ?
          ]
     ]
     if :direction = 3 [
          if (last ?) <= :s [
               make "change_direction "false
               make "iduci ?
          ]
     ]
     if :direction = 4 [
          if (first ?) <= :r [
               make "change_direction "false
               make "iduci ?
          ]
     ]
]
ifelse :change_direction [
     lt 90
```

```
                make "direction :direction+1
          ] [
                ; pomak
                make "poz pos
                pu rt 90 fd :a lt 90 fd :a pd
                setpos :poz
                pu rt 90 fd :a lt 90 fd :a pd
                if :direction = 1 [
                     make "r :r+1 ;Dijagonala gore desno
                     make "s :s+1
                ]
                if :direction = 2 [
                     make "r :r+1 ;Dijagonala gore lijevo
                     make "s :s-1
                ]
                if :direction = 3 [
                     make "r :r-1 ;Dijagonala dolje lijevo
                     make "s :s-1
                ]
                if :direction = 4 [
                     make "r :r-1 ;Dijagonala dolje desno
                     make "s :s+1
                ]
          ]
     ]
     rt 90 fd :a*((last :min_r)-:s) lt 90
end

to kvadrat
     repeat 4 [fd :a rt 90]
     pu fd 3 rt 90 fd 3 pd
     fill
     pu bk 3 lt 90 bk 3 pd
end

to nacrtaj_kvadrate
     setpc 0
     make "poz pos
     foreach :l [
          pu rt 90 fd ((last ?)-1)*:a lt 90 fd ((first ?)-1)*:a pd
          kvadrat
          pu bk ((first ?)-1)*:a rt 90 bk ((last ?)-1)*:a lt 90 pd
     ]
end
```