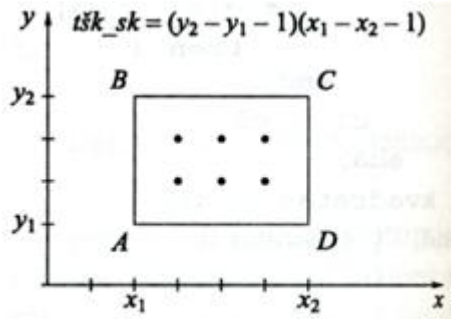
Stačiakampis

Šiame uždavinyje daugiau matematikos, nei programavimo, be to, jį galima spręsti skirtingais metodais.

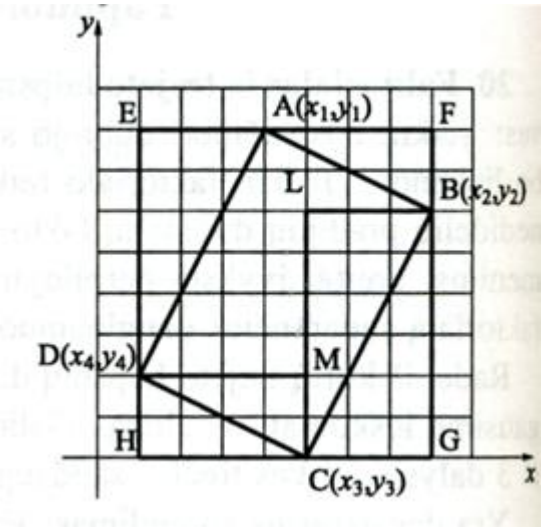
Nors stačiakampį galima vienareikšmiškai nusakyti trijų kampų koordinatėmis, laikome, kad yra duotos visos keturios. Jos išvardytos pagal laikrodžio rodyklę.

Akivaizdu, kad jei stačiakampio kraštinės yra lygiagrečios koordinačių ašims, taškų kiekį stačiakampio viduje rasti labai paprasta



21.1 pav.

Kai stačiakampis yra pasviręs koordinačių ašių atžvilgiu, rasti taškų kiekį yra sunkiau. Papildykime duotąjį stačiakampį iki koordinačių ašims lygiagretaus stačiakampio



Taškai, esantys trikampių AED ir BCG (jie yra lygūs) viduje nepriklauso stačiakampiui ABCD. Jų skaičius lygus skaičiui taškų, esančių stačiakampio BLCG viduje:

$$\text{tšk_sk} = (y_2 - y_3 - 1) \times (x_2 - x_3 - 1)$$

Be to, reikia sužinoti, ar atkarpoje AD (taškus, esančius atkarpoje BC, jau atmetėme) yra ieškomų taškų (jie nėra stačiakampio ABCD viduje, todėl juos reikia pridėti prie stačiakampio BLCG vidinių taškų).

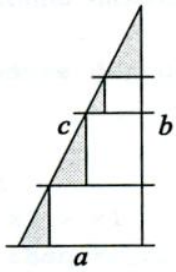
Taškų, kuriuos kerta atkarpa AD, skaičius lygus atkarpų ED ir EA didžiausiam bendrajam dalikliui minus 1.

Analogiškai suskaičiuojamas ir trikampiuose AFB ir DHC esančių taškų skaičius: randamas taškų skaičius stačiakampio DHCM viduje ir papildomas taškų, esančių atkarpose AB ir CD, skaičiumi.

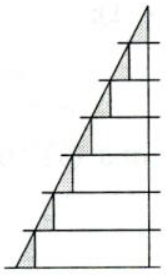
Ieškomasis stačiakampio ABCD viduje esančių taškų skaičius bus gautas iš stačiakampio EFGH vidinių taškų skaičiaus atėmus stačiakampių BLCG ir DHCM taškus.

Šiame algoritme remiamasi teiginiu, kad s*tačiojo trikampio įžambinei priklausančių taškų su sveikosiomis koordinatėmis skaičius lygus abiejų statinių didžiausiam bendrajam dalikliui minus 1, jei neįskaitomi įžambinės galai*.

Įrodysime šį teiginį. Imkime bet kokį statųjį trikampį, kurio statinių ilgiai yra sveikieji skaičiai. Nubrėžkime keletą trikampio pagrindui lygiagrečių atkarpų. Tose vietose, kur atkarpos kerta įžambinę *c*, nuleiskime aukštines į trikampio pagrindą. Taip gausime keletą prigludusių prie įžambinės mažų trikampių



21.3 pav.



21.4 pav.

Visi mažieji trikampiai bus panašūs į didįjį trikampį (jų visi atitinkami kampai lygūs).

Dabar pakoreguokime tuos mažus trikampius: juos nupieškime taip, kad visi mažieji trikampiai būtų vienodi. Taip bus tik tada, jei trikampio vertikalųjį statinį padalysime į lygias dalis ir iš įstrižainę kertančių taškų nuleisime aukštines. Kadangi trikampiai yra panašūs, tai, jei sutampa vienas iš statinių, turi sutapti ir kitų kraštinių ilgiai

Norint, kad trikampių vertikalių statinių ilgiai būtų sveikieji skaičiai, didžiojo trikampio statinį b reikia dalyti į k dalių, čia k – bet koks b daliklis. Antrasis mažųjų trikampių statinis bus sveikasis skaičius tik tada, jei k bus ir a daliklis.

Abu kiekvieno mažojo trikampio įstrižainės galai priklauso įstrižainei c. Jei kiekvieno jų abiejų statinių ilgai sveikieji skaičiai, tai ir taškų, esančių abiejuose įstrižainės galuose, koordinatės bus sveikieji skaičiai.

Įsitikinome, kad jei trikampį galima aukščiau paaiškintu būdu padalyti į mažesnius trikampius, kurių statinių ilgiai yra sveikieji skaičiai, tai mažesnių trikampių įstrižainių galai priklausys pradinio trikampio įstrižainei ir jų koordinatės bus sveikieji skaičiai.

Kiek daugiausiai tokių taškų gali būti? Tiek, kiek yra mažųjų trikampių minus 1. Na o jų daugiausiai gali būti tiek, koks yra statinių a ir b didžiausias bendrasis daliklis.

Nors sprendimo idėjos aprašas ir įrodymas gana ilgi, pats algoritmas – trumpas:

```
type taškas = record
                x, y: integer;
              end;

function dbd (x, y: integer): integer;
{ randa skaičių x ir y didžiausiąjį bendrą daliklį }
begin
  if x = 0
     then dbd := y
     else dbd := dbd (y mod x, x)
end;   { dbd }

function max (x, y: integer): integer;
begin
  if x > y
     then max := x
     else max := y
end;

function taškai (a, b, c, d: taškas): integer;
{ randa stačiakampio viduje esančių taškų skaičių }
  var ilgis, plotis, { stačiakampio dydis }
      a1, b1, a2, b2, { trikampių statiniai }
      tašk, tašk1, tašk2: integer;
begin
  { randame gaubiančiojo stačiakampio dydį }
  plotis := max (abs (a.x - c.x), abs (b.x - d.x));
  ilgis := max (abs (a.y - c.y), abs (b.y - d.y));
  tašk := (plotis - 1) * (ilgis - 1);
  { rasime pirmo trikampio kraštines }
  a1 := abs (a.x - b.x);
  b1 := abs (a.y - b.y);
  tašk1 := (a1  - 1) * (b1 - 1) + dbd (a1, b1) - 1;
  { taip pat ir antro trikampio }
  a2 := abs (b.x - c.x);
  b2 := abs (b.y - c.y);
  tašk2 := (a2 - 1) * (b2 - 1) + dbd (a2, b2) - 1;
  taškai := tašk - tašk1 - tašk2
end;
```