Računska geometrija za tekmovalno programiranje

Vid Kocijan

University of Oxford, Department of Computer Science

vid.kocijan@cs.ox.ac.uk

april 2020

Pregled vsebine ali vektorski produkt na 1001 način

- osnovne strukture
- ploščine
- sortiranje po kotu
- konveksne ovojnice

Predstavitev osnovnih objektov

Kjer se da, se ognemo decimalkam.

- Točka: par (x, y), razdalja med točkama $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Če potrebujemo zgolj za primerjavo, korena ne računamo.

Predstavitev osnovnih objektov

Kjer se da, se ognemo decimalkam.

- Točka: par (x, y), razdalja med točkama $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Če potrebujemo zgolj za primerjavo, korena ne računamo.
- Daljica: par točk (T_1, T_2). Kako preveriti, ali točka U leži na daljici? $\vec{U} = \alpha \vec{T}_1 + (1 - \alpha)\vec{T}_2, \alpha \in [0, 1]$

Predstavitev osnovnih objektov

Kjer se da, se ognemo decimalkam.

- Točka: par (x, y), razdalja med točkama $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Če potrebujemo zgolj za primerjavo, korena ne računamo.
- Daljica: par točk (T_1, T_2). Kako preveriti, ali točka U leži na daljici? $\vec{U} = \alpha \vec{T}_1 + (1 - \alpha) \vec{T}_2, \alpha \in [0, 1]$
- Trikotnik: trojica točk v pozitivni smeri.
- Mnogokotnik: seznam točk v pozitivni smeri.
- Krog: središče S in radij r. Preverjanje, ali je točka X v krogu: $d(X, S)^2 \leq r^2$.

Predstavitev osnovnih objektov

Kjer se da, se ognemo decimalkam.

- Točka: par (x, y), razdalja med točkama $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Če potrebujemo zgolj za primerjavo, korena ne računamo.
- Daljica: par točk (T_1, T_2). Kako preveriti, ali točka U leži na daljici? $\vec{U} = \alpha \vec{T}_1 + (1 - \alpha)\vec{T}_2, \alpha \in [0, 1]$
- Trikotnik: trojica točk v pozitivni smeri.
- Mnogokotnik: seznam točk v pozitivni smeri.
- Krog: središče S in radij r. Preverjanje, ali je točka X v krogu: $d(X, S)^2 \leq r^2$.
- Premica: trojica (a, b, c), predstavlja $ax + by = c$. Pri računanju **pazimo na deljenje z 0**.

Ko stvari niso celoštevilske

Problem računske natančnosti:

$$(\frac{3 - 0.001 \cdot 3}{3} - 1) \cdot 1000 + 1 = 0$$

Ko stvari niso celoštevilske

Problem računske natančnosti:

$$\left(\frac{3 - 0.001 \cdot 3}{3} - 1\right) \cdot 1000 + 1 = 0$$

Python se ne strinja:

```
>>> ((3-(0.001*3))/3-1)*1000+1
-8.881784197001252e-16
>>> ((3-(0.001*3))/3-1)*1000+1==0
False
```

Ko stvari niso celoštevilske

Problem računske natančnosti:

$$(\frac{3 - 0.001 \cdot 3}{3} - 1) \cdot 1000 + 1 = 0$$

Python se ne strinja:

```
>>> ((3-(0.001*3))/3-1)*1000+1
-8.881784197001252e-16
>>> ((3-(0.001*3))/3-1)*1000+1==0
False
```

Rešitev: `const double eps = 1e-9;`

`a==b` $\rightarrow$ `fabs(a-b)<=eps`

Ko stvari niso celoštevilske

Problem računske natančnosti:

$$(\frac{3-0.001\cdot 3}{3}-1)\cdot 1000+1=0$$

Python se ne strinja:

```
>>> ((3-(0.001*3))/3-1)*1000+1
-8.881784197001252e-16
>>> ((3-(0.001*3))/3-1)*1000+1==0
False
```

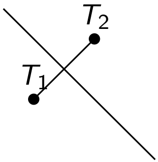
Rešitev:`const double eps = 1e-9;`

`a==b` $\rightarrow$ `fabs(a-b)<=eps`

Bisekcija: `while(d-l>eps)` ali pa kar `for(int i=0;i<100;i++)`

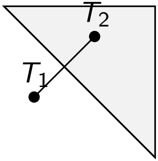
Presečišče premice in daljice

Imamo premico $ax + by + c = 0$ in daljico med $T_1 = (x_1, y_1)$ in $T_2 = (x_2, y_2)$.



Presečišče premice in daljice

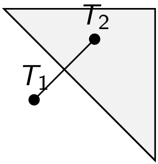
Imamo premico $ax + by + c = 0$ in daljico med $T_1 = (x_1, y_1)$ in $T_2 = (x_2, y_2)$.



Preverjanje na kateri strani premice je točka: $ax_1 + by_1 + c > 0$.

Presečišče premice in daljice

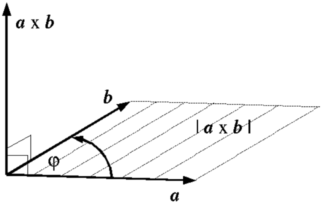
Imamo premico $ax + by + c = 0$ in daljico med $T_1 = (x_1, y_1)$ in $T_2 = (x_2, y_2)$.



Preverjanje na kateri strani premice je točka: $ax_1 + by_1 + c > 0$.
Iskanje presečišča: $\vec{S} = \alpha \vec{T}_1 + (1 - \alpha)\vec{T}_2, \alpha \in [0, 1]$, bisekcija po α.
Ognemo se morebitni računski napaki, ki bi nastala, če bi presečišče računali z eksplicitno formulo.

Vektorski produkt

Ponovimo srednješolsko matematiko za našim glavim orodjem.



$$\vec{a} \times \vec{b} = (0, 0, a_1 b_2 - b_1 a_2)$$

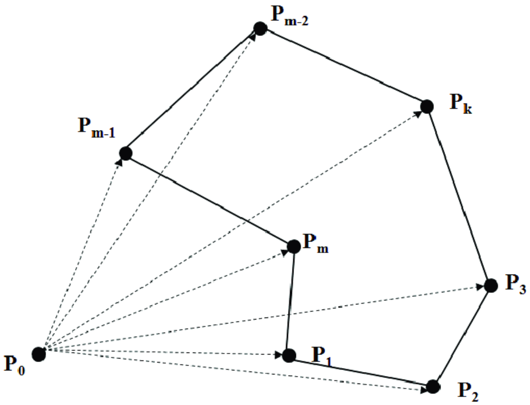
- Kolinearnost treh točk A, B, C? $\vec{AB} \times \vec{AC} = 0$

- Kolinearnost treh točk A, B, C? $\vec{AB} \times \vec{AC} = 0$
- Orientacija trikotnika A, B, C? $\vec{AB} \times \vec{AC} \geq 0$

- Kolinearnost treh točk A, B, C? $\vec{AB} \times \vec{AC} = 0$
- Orientacija trikotnika A, B, C? $\vec{AB} \times \vec{AC} \geq 0$
- Ploščina trikotnika A, B, C? $\frac{\vec{AB} \times \vec{AC}}{2}$

- Kolinearnost treh točk A, B, C? $\vec{AB} \times \vec{AC} = 0$
- Orientacija trikotnika A, B, C? $\vec{AB} \times \vec{AC} \geq 0$
- Ploščina trikotnika A, B, C? $\frac{\vec{AB} \times \vec{AC}}{2}$
- Pravokotnost $\vec{a}$ in $\vec{b}$? Skalarni produkt $\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2 = 0$

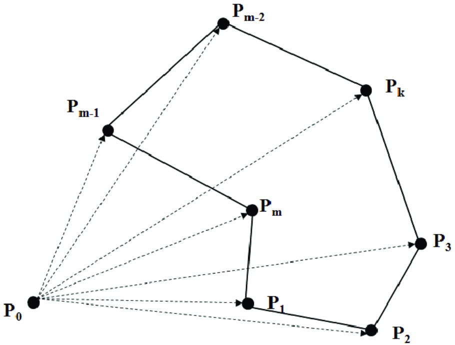
Ploščina mnogokotnika



$$p = \sum_{i=1}^{n} \frac{\vec{P_i} \times \vec{P_{i+1}}}{2}$$

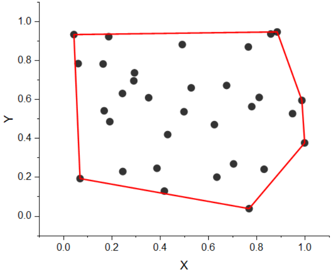
Ne deluje za samo-sekajoče like.

Sortiranje po kotu



Če so koti manjši od π gledamo orientacijo trikotnika (P_0, P_i, P_j).
Če so koti večji od π dodamo kup if stavkov.

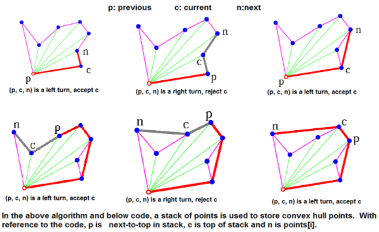
Konveksnost



Algebraična definicija: Imamo točke $T_1, \dots, T_n$. X je v konveksni ovojnici, če velja

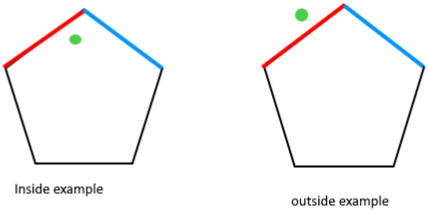
$$\vec{X} = \sum_{i=1}^{n} \alpha_i \vec{T}_i; \quad \sum_{i=1}^{n} \alpha_i = 1; \quad \alpha_i \in [0, 1]$$

Graham Scan



Točke uredimo po kotu od najbolj levo spodnje. Nato dodajamo na sklad, ki predstavlja našo trenutno ovojnico. Ko na sklad vzamemo novo točko gledamo orientacijo trikotnika iz zadnjih 3 točk. Če smo ,,zavili v desno'', odstranimo predzadnjo točko. Ponavljamo odstranjevanje dokler zadnje 3 točke na skladu niso orientirane pozitivno. Zahtevnost $O(n \log n)$.

Kako preverimo, ali je neka točka znotraj konveksne ovojnice?



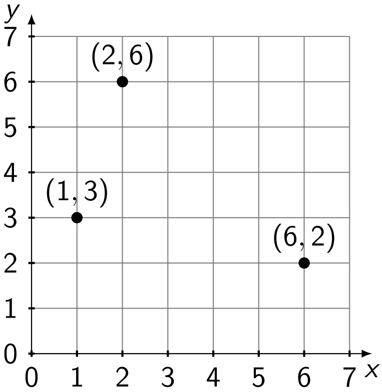
Točka P je v mnogokotniku $T_1, \ldots, T_n$, če je trikotnik (T_i, T_{i+1}, P) orientiran pozitivno za vsak i.

Primer

Naloga s Codeforces: Freelancer's Dreams, 605/C

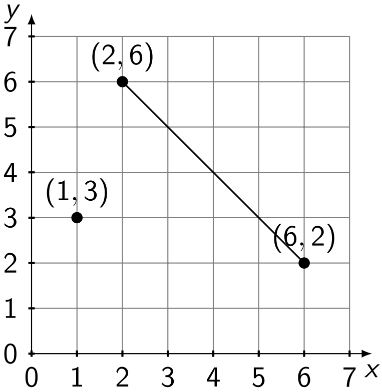
Primer

Naloga s Codeforces: Freelancer's Dreams, 605/C



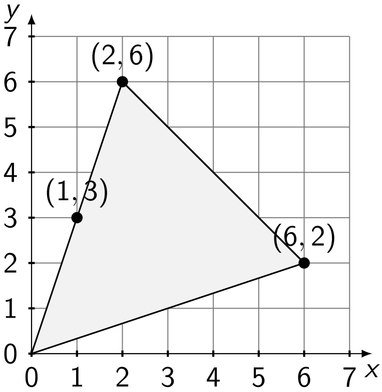
Primer

Naloga s Codeforces: Freelancer's Dreams, 605/C

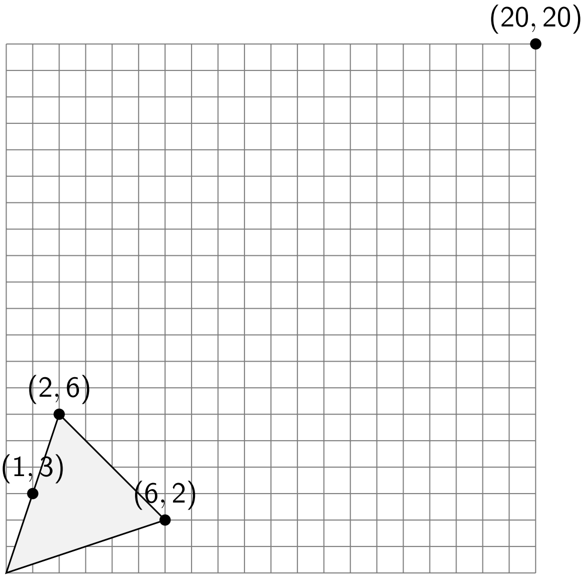


Primer

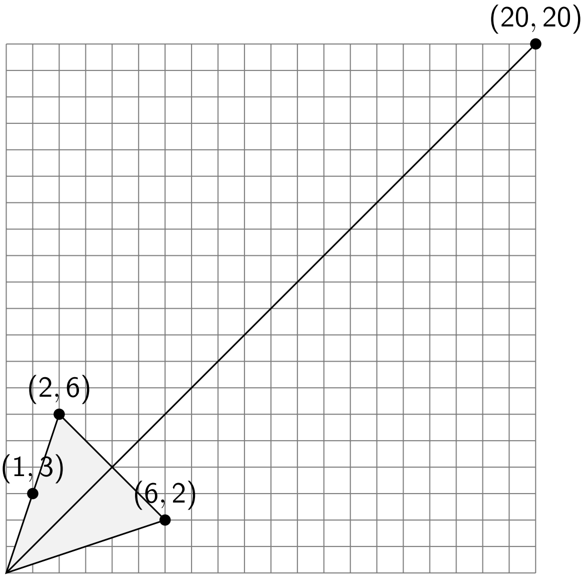
Naloga s Codeforces: Freelancer's Dreams, 605/C



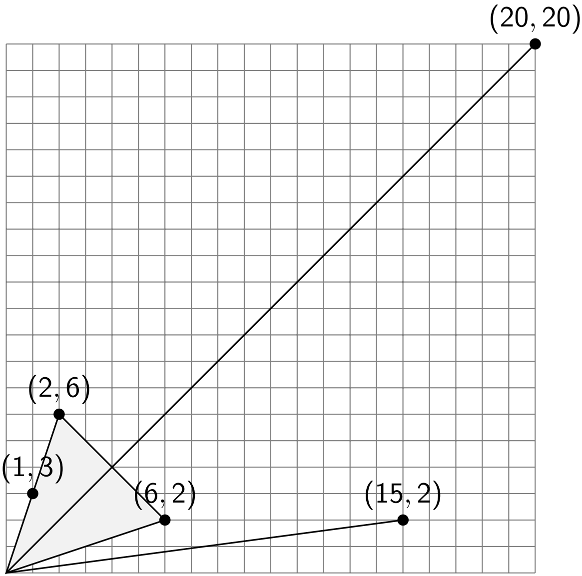
Primer



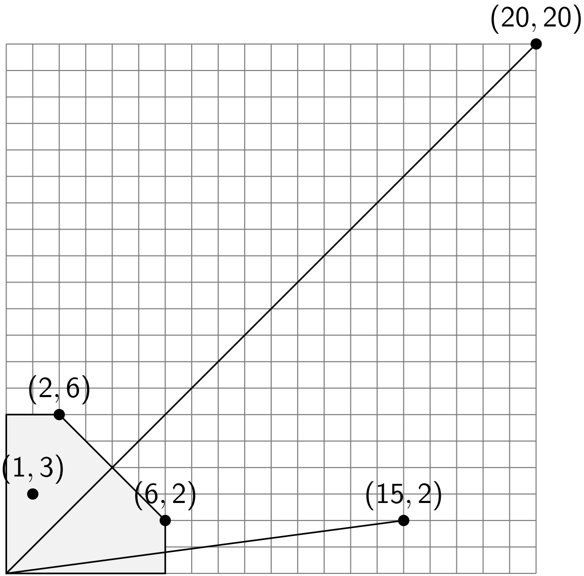
Primer



Primer



Primer



Povzetki

- Vektorski produkt je zakon.
- Pazi na deljenje z 0.
- Pazi na računsko napako pri decimalkah.
- Ogibaj se decimalnih števil ko le lahko.
- Vektorski produkt je zakon.