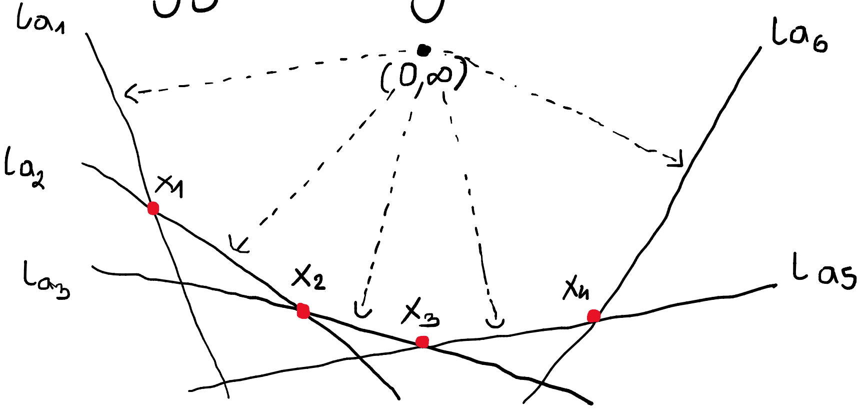
Zad 2.

niedziela, 16 kwietnia 2023 20:29

2. (2pkt) Danych jest n prostych $l_1, l_2, \ldots l_n$ na płaszczyźnie ($l_i = a_i x + b_i$), takich że żadne trzy proste nie przecinają się w jednym punkcie. Mówimy, że prosta l_i jest widoczna z punktu p jeśli istnieje punkt q na prostej l_i, taki że odcinek $\overline{pq}$ nie ma wspólnych punktów z żadną inną prostą l_j ($j \neq i$) poza (być może) punktami p i q.

Ułóż algorytm znajdujący wszystkie proste widoczne z punktu (0,+∞).

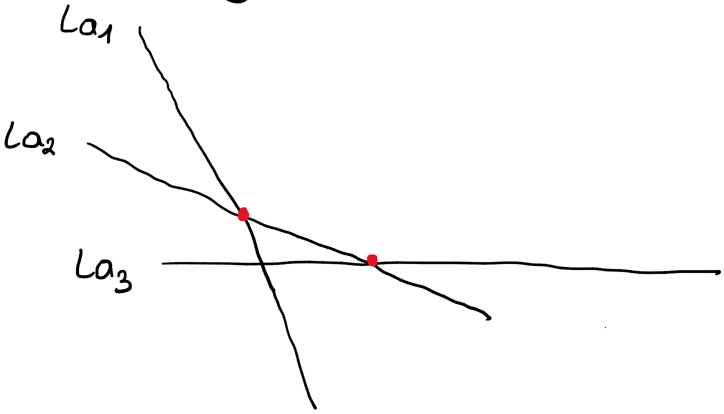
najpierw popatrzmy sobie na to jak mniej więcej będzie wyglądał wynik:



+ prosta obserwacja: $\{x_i\}_{i=1}$ jest rosnący

1) zaczynamy od posortowania prostych względem współczynnika kierunkowego
(spośród tych prostych, które są do siebie, równoległe, zostawiamy te o najw. współczynniku b_i

2) będziemy iterować się po, posortowanym ciągu i stopniowo tworzyć zbiór wynikowy.
popatrzmy sobie na częściowe rozwiązanie:

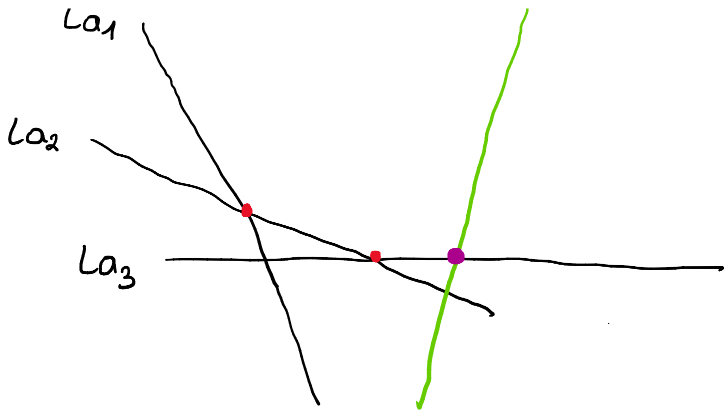


rozpatrujemy zieloną prostą
(2 przypadki)

a)

rozpatrywana prosta nic nie zasłania

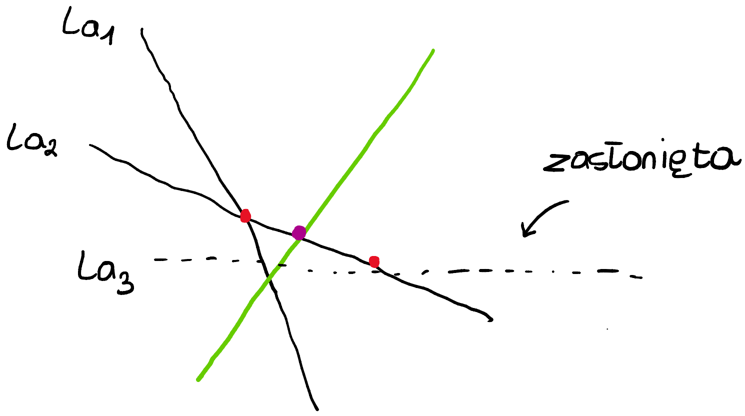
a)



prosta nic nie zasłania (git, po prostu dodajemy ją do wyniku)

b) zielona prosta przysłania co najmniej jedną prostą wynikową

(dodajemy ją do wyniku i usuwamy przysłonięte proste)



Q: jak usuwamy?

potrzebujemy funkcji liczącej x-ową współrzędną przecięcia dwóch prostych.

($ax+b=cx+d \Rightarrow x=\frac{b-d}{c-a}$)

w trakcie programu pamiętamy ciąg $\{x_i\}_{i=1}$ (x_i odpowiada prostym La_i oraz La_{i+1})

przez to, że $\{x_i\}_{i=1}$ jest monotoniczny, to możemy na nim wykonać wyszukiwanie binarne → szukamy najmniejszego takiego x_k, że intersect(l_j, l_k) $<$ x_k. Usuwamy z wyniku wszystkie proste La_i dla $i \geq k+1$, dostosowujemy również $\{x_i\}$

można też szybciej idąc od prawej, bo

można też szybciej idąc od prawej, bo korzystamy z obserwacji, że usuwamy spójny sufiks