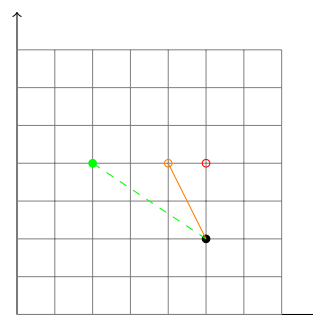
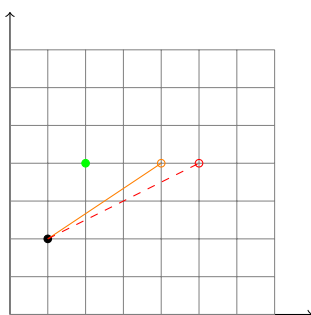


Z podanych punktów $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ musimy wybrać taką parę (i, j) , żeby nachylenie odcinka $i \rightarrow j$, czyli wartość:

$$\frac{|y_i - y_j|}{|x_i - x_j|}$$

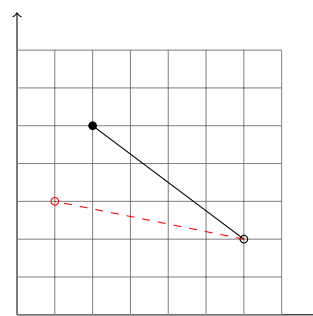
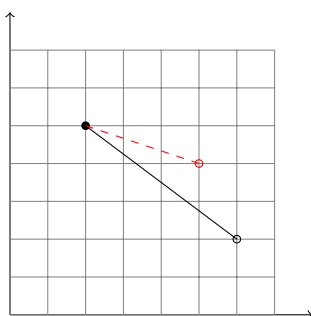
była najmniejsza możliwa. Na pierwszy rzut oka wydaje się, że musimy sprawdzić $O(N^2)$ par punktów, a więc nie zejdziemy poniżej tej złożoności. Szybko jednak okazuje się, że nie musimy sprawdzać wszystkich par, bo kandydatów na najlepszą jest znacznie mniej. Zapewniają nam to następujące dwie obserwacje:

1. Jeśli dla pewnej wartości y są przynajmniej trzy punkty z tą wartością: $(x_i, y), (x_j, y), (x_k, y)$ przy $x_i < x_j < x_k$, to punkt środkowy (x_j, y) nie może stworzyć najlepszej pary z żadnym innym punktem.



Na powyższym rysunku są trzy punkty (czerwony, pomarańczowy i zielony) o tej samej współrzędnej y . Gdziekolwiek znajduje się czarny punkt, odcinek między czarnym i pomarańczowym ma zawsze większe nachylenie albo od odcinka czarny-czerwony, albo od odcinka czarny-zielony.

2. Dla dwóch punktów (x_i, y_i) oraz (x_j, y_j) takich, że $y_i < y_j$, jeśli istnieje trzeci punkt $A_k = (x_k, y_k)$ taki, że $y_i < y_k < y_j$, to odcinek $A_i A_j$ nie może mieć najmniejszego nachylenia.



Odcinek między czarnymi punktami nie może mieć najlepszego nachylenia – czerwony punkt o współrzędnej y pomiędzy czarnymi powoduje, że jeden z czerwonych odcinków zawsze będzie lepszy.

Posortujmy dane punkty względem współrzędnej y , czyli ustawmy w kolejności $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ takiej, że $y_1 \leq y_2 \leq \dots \leq y_N$. Punkty o równej współrzędnej y niech z kolei będą uporządkowane względem współrzędnej x – zarówno sortowanie par w C++ (typ `pair<int, int>` z biblioteki STL), jak i sortowanie par w Pythonie zapewnia nam taki porządek bez konieczności pisania dodatkowego kodu.

Punkty o tej samej współrzędnej y tworzą teraz grupy sąsiednich elementów – w każdej grupie może być jeden lub więcej punktów. Z obserwacji 1. wynika, że z każdej takiej grupy wystarczy pozostawić pierwszy i ostatni punkt (tj. o najmniejszej i największej współrzędnej x), bo żaden z pozostałych nie jest kandydatem do najlepszego odcinka. Z kolei

obserwacja 2. gwarantuje, że do znalezienia najlepszej pary wystarczy sprawdzić punkty z sąsiadujących grup. Algorytm rozwiązujący nasze zadanie może więc wyglądać następująco:

- Posortuj punkty względem współrzędnej y (remisy rozstrzygając względem współrzędnej x) i podziel na grupy o równym y .
- Usuń z każdej grupy wszystkie punkty poza pierwszym i ostatnim.
- Dla każdego dwóch punktów (x_i, y_i) oraz (x_j, y_j) należących do sąsiednich grup sprawdź wartość nachylenia $\frac{|y_i - y_j|}{|x_i - x_j|}$ i zapamiętaj najmniejszą ze znalezionych wartości.

Skoro w każdej grupie pozostają co najwyżej dwa punkty, to dla każdego dwóch sąsiednich grup będzie trzeba sprawdzić co najwyżej 4 pary, a zatem łącznie jest co najwyżej $4 \cdot N$ par-kandydatów – tym samym nasz algorytm działa teraz w czasie $O(N)$.

