JOI 2025/26 ファイナル Day 4

パン職人 (Baker)

解説：渡邉雄斗 (yuto1115)

問題概要

- M 人の客がいる　客 i は時刻 T_i にクロワッサンを注文し、

  最大で時刻 $T_i + L$ まで待てる

- Q 個のクエリがある　各クエリでは A_q, B_q が与えられる

  - A_q 分でクロワッサンを作れる職人が 1 人、時刻 B_q に出勤
  - **出勤時刻までに入った注文のみ**考慮する
  - 最大で何人の客の注文を捌けるか？

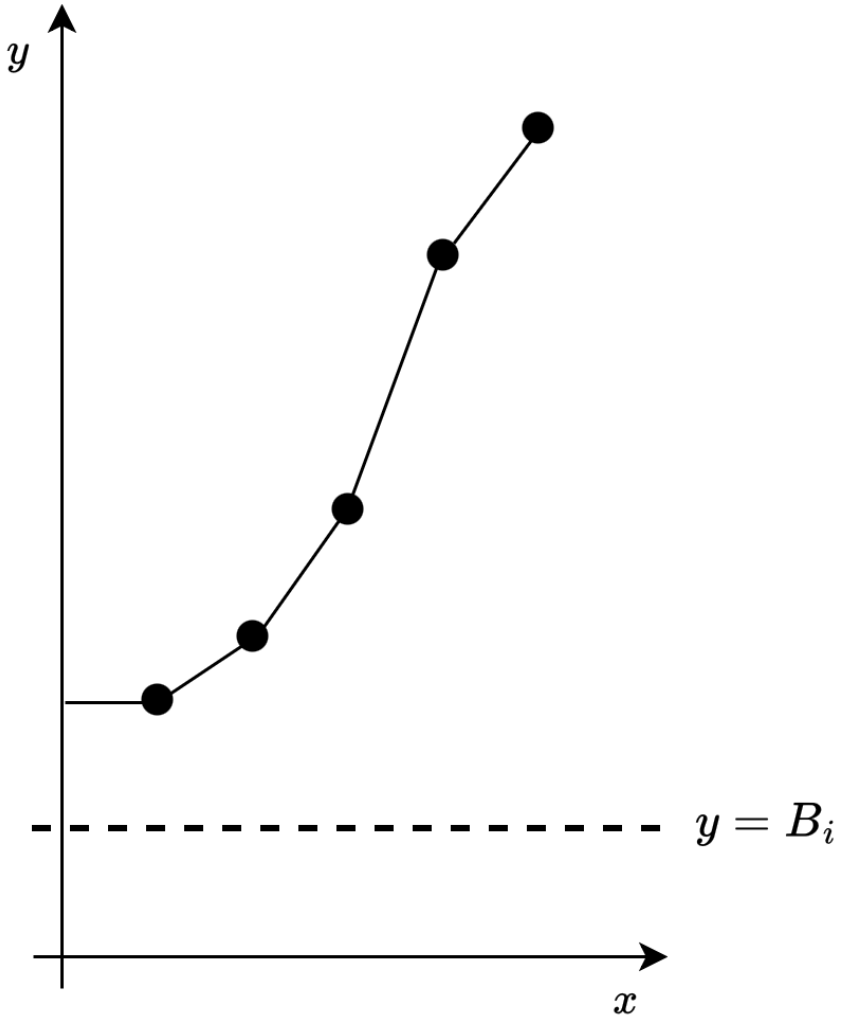
- 制約：$M \leq 2\times10^6,\ Q \leq 4\times10^5$

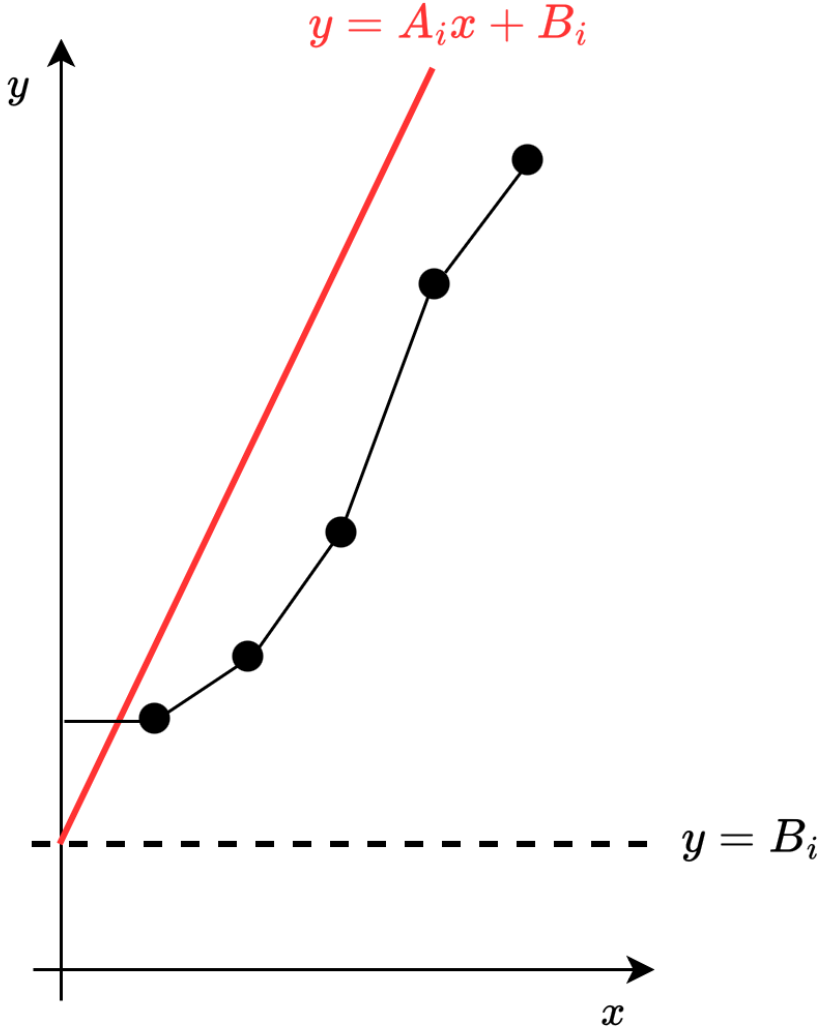
Subtask 1：$M \leq 10,\ Q \leq 10^5$

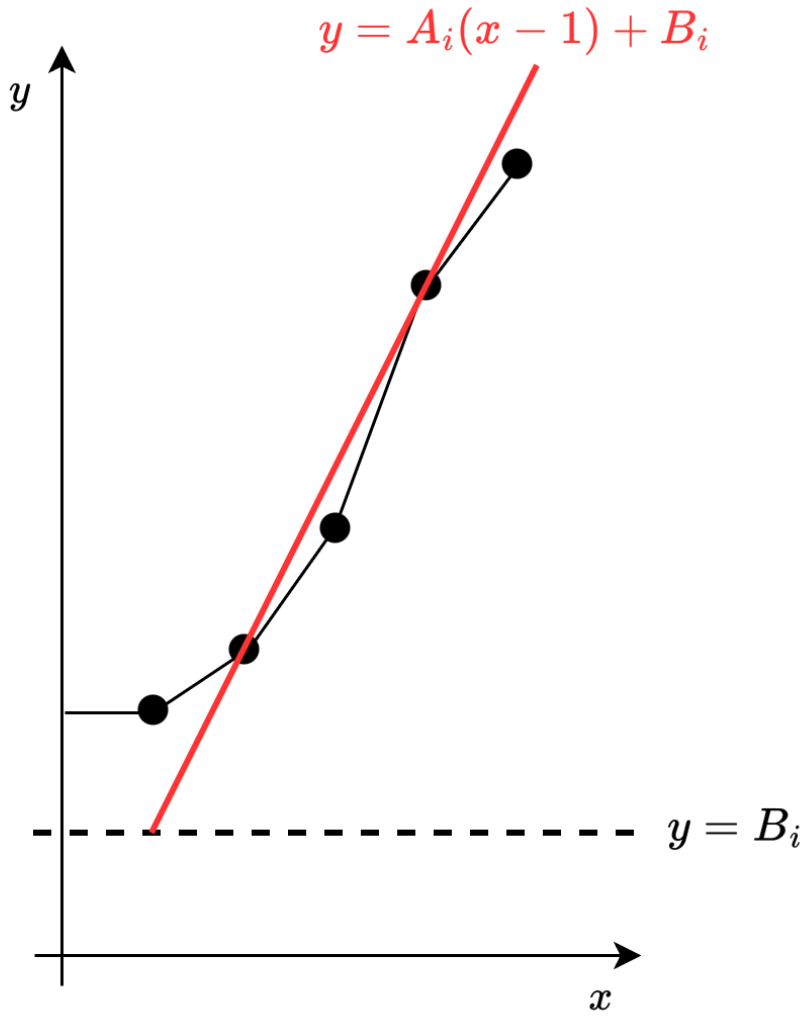
- 以下の値を導入します
  - $Y_i\ (1 \leq i \leq M) \coloneqq T_i + L$　：客 i の締切時刻
  - $L_q, R_q\ (1 \leq q \leq Q)$ ：クエリ q が考慮する客の区間 $[L_q, R_q)$
    - $T_i \leq B_q \leq Y_i$ なる客が考慮される
- $k = 1, 2, \ldots, R_q - L_q$ それぞれについて、k 人の客を捌けるか判定したい
  - $R_q - k, R_q - k + 1, \ldots, R_q - 1$ をこの順に貪欲に捌くのが最適
- $O(QM^2)$ 時間

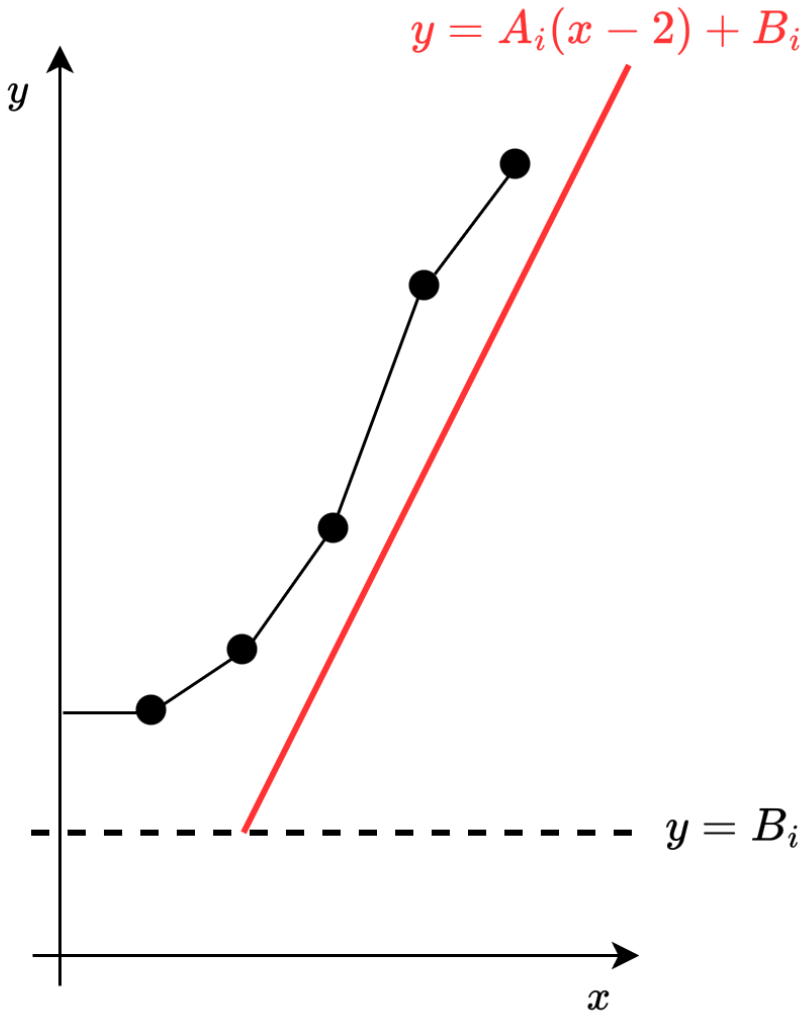
Subtask 2 : $M \leq 500,\ Q \leq 10^5$

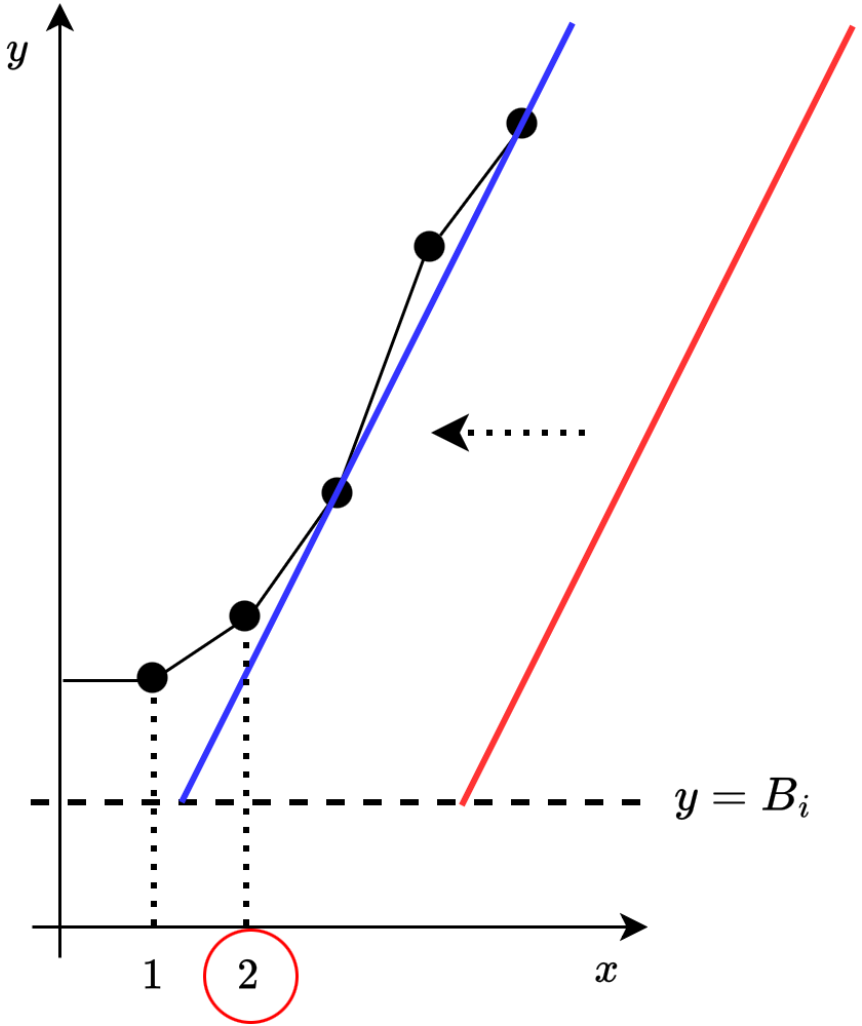
- 前から貪欲、Sub 1 + 二分探索など色々あるが、ここでは次につながる解法を紹介
- $k = 1, 2, \ldots, R_q - L_q$ それぞれについて、k 人の客を捌けるか判定したい
  - $R_q - k, R_q - k + 1, \ldots, R_q - 1$ をこの順に貪欲に捌くのが最適
  - $\Leftrightarrow Y_i \geq B_q + A_q(i - R_q + k + 1)\ (R_q - k \leq i < R_q)$
  - $\Leftrightarrow Y_i \geq B_q + A_q(i - R_q + k + 1)\ \ (L_q \leq i < R_q)$
  - $\Leftrightarrow k \leq \frac{Y_i - B_q}{A_q} + R_q - 1 - i\ \ \ (L_q \leq i < R_q)$
- 右辺の最小値の floor が答え　$O(QM)$ 時間

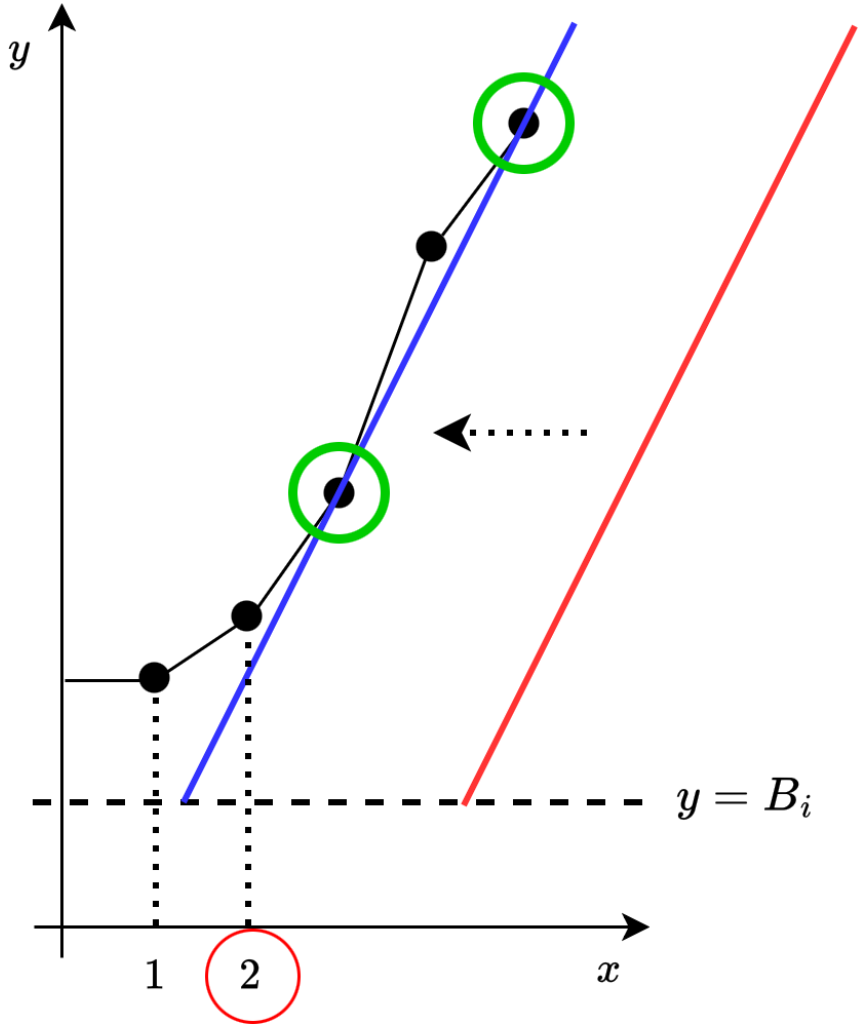

y
x
$y = B_i$


$y = A_i x + B_i$
y
$y = B_i$
x


$y = A_i(x-1) + B_i$
y
$y = B_i$
x


$y = A_i(x-2) + B_i$
y
$y = B_i$
x


y
x
1
2
$y = B_i$


y
x
1
2
$y = B_i$

Subtask 2 : $M \leq 500,\ Q \leq 10^5$

- グラフを使って考える
  - M 個の点 $(1, Y_1), (2, Y_2), \ldots, (M, Y_M)$ がある（(i, Y_i) を点 i と呼ぶことにする）
- 捌きたい客の人数 k を固定すると、

  それぞれの客に対するサーブ時刻は傾き A_q の直線で表される（k 小さいほど右に位置）
  - 点 $[L_q,\ R_q)$ が全てこの直線の上（境界含む）にあることが必要十分
- 言い換えると、傾き A_q の直線を右から動かした時、いつぶつかるか知りたい
- $-A_q x + y$ を最小にする点を見つければ、その点の情報から $O(1)$ で求解

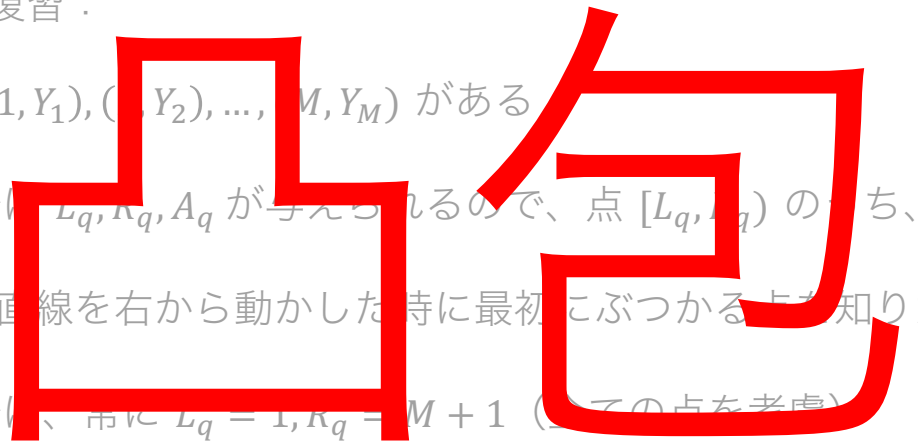
Subtask 3 : $T_M \leq B_q < T_1 + L$

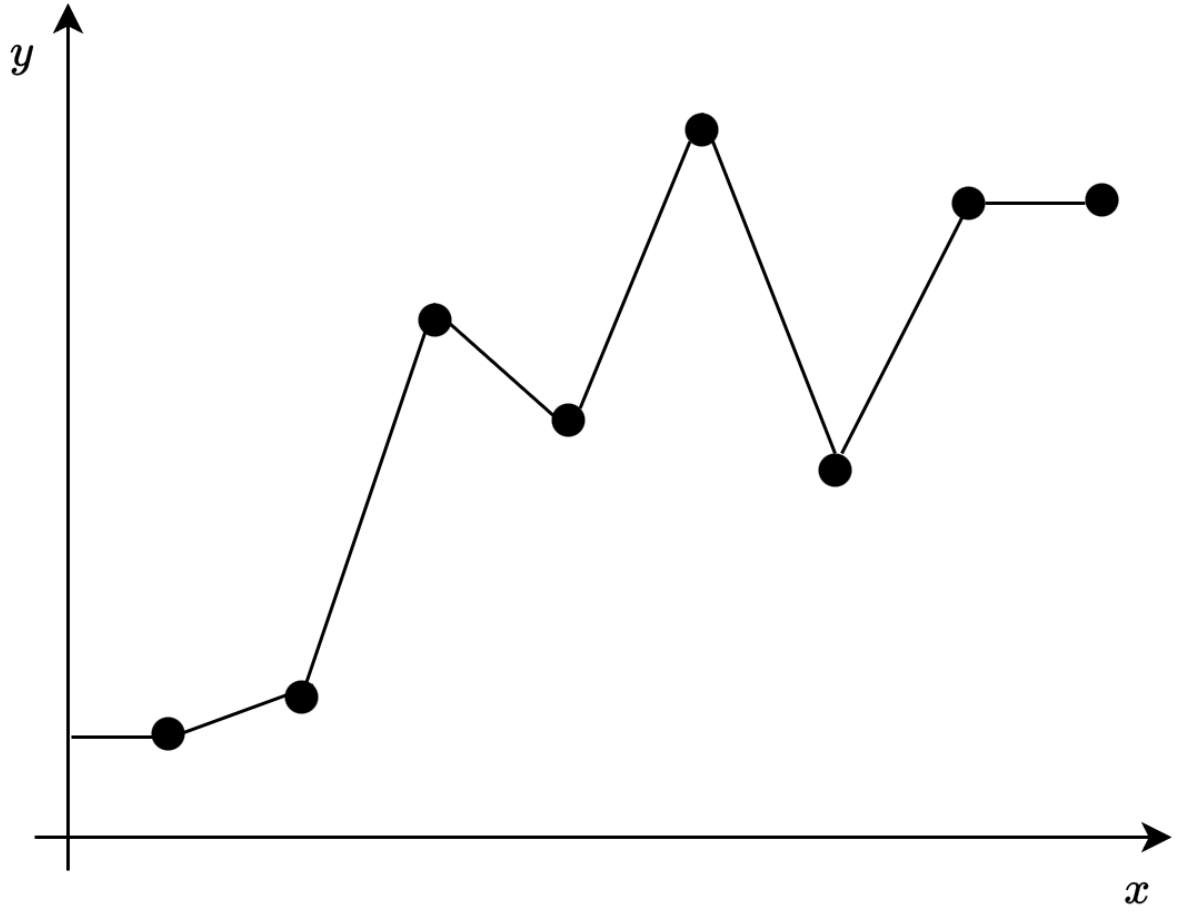
- 帰着後の問題の復習：
  - M 個の点 $(1, Y_1), (2, Y_2), \ldots, (M, Y_M)$ がある
  - 各クエリでは L_q, R_q, A_q が与えられるので、点 $[L_q, R_q)$ のうち、

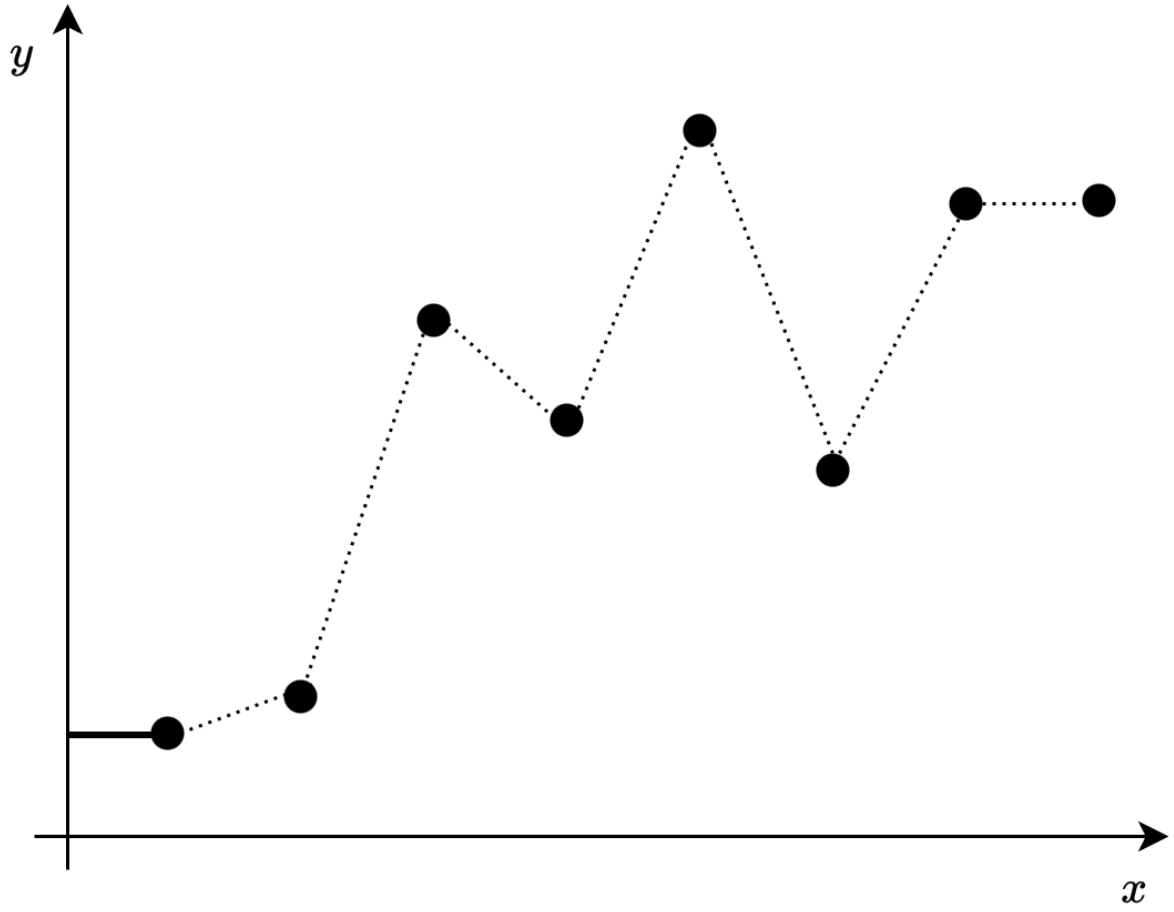
    傾き A_q の直線を右から動かした時に最初にぶつかる点を知りたい
  - 本小課題では、常に $L_q = 1, R_q = M + 1$（全ての点を考慮）

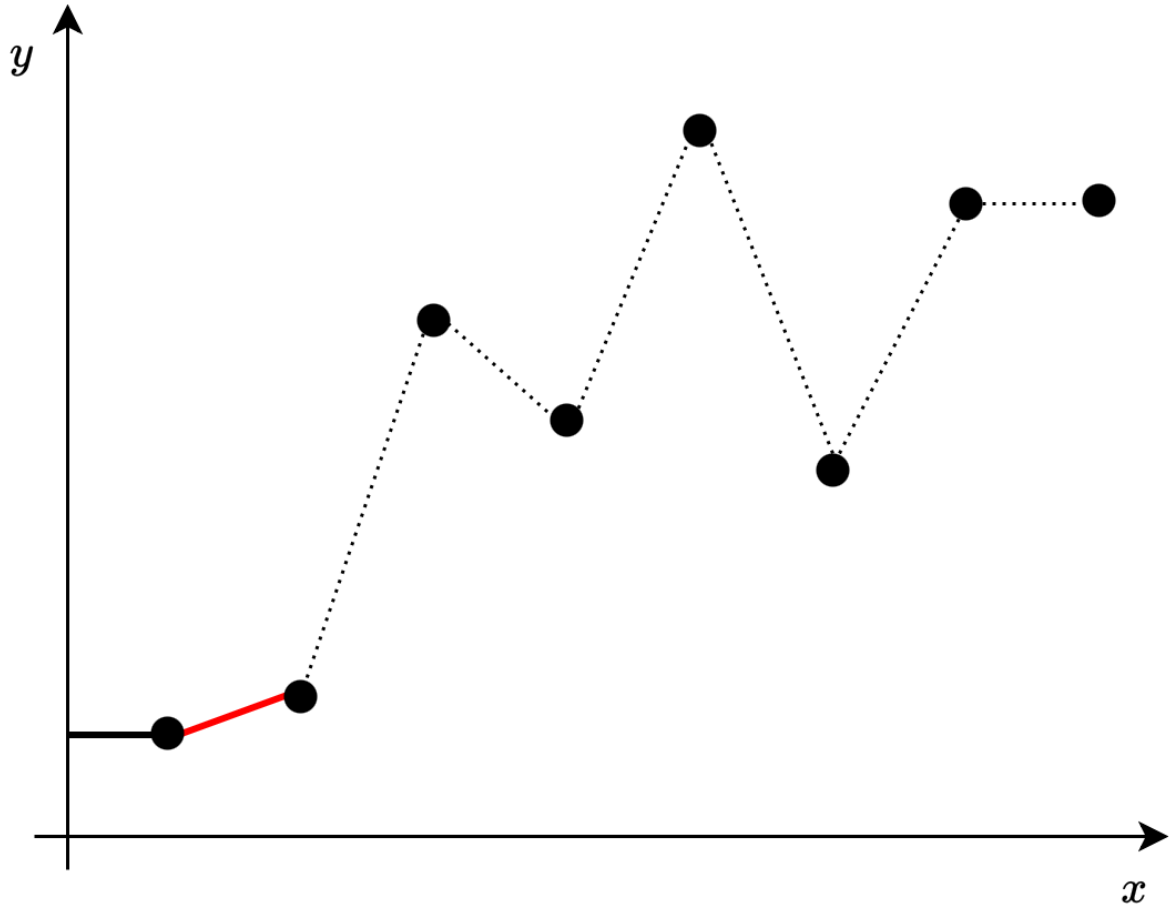
Subtask 3 : $T_M \leq B_q < T_1 + L$

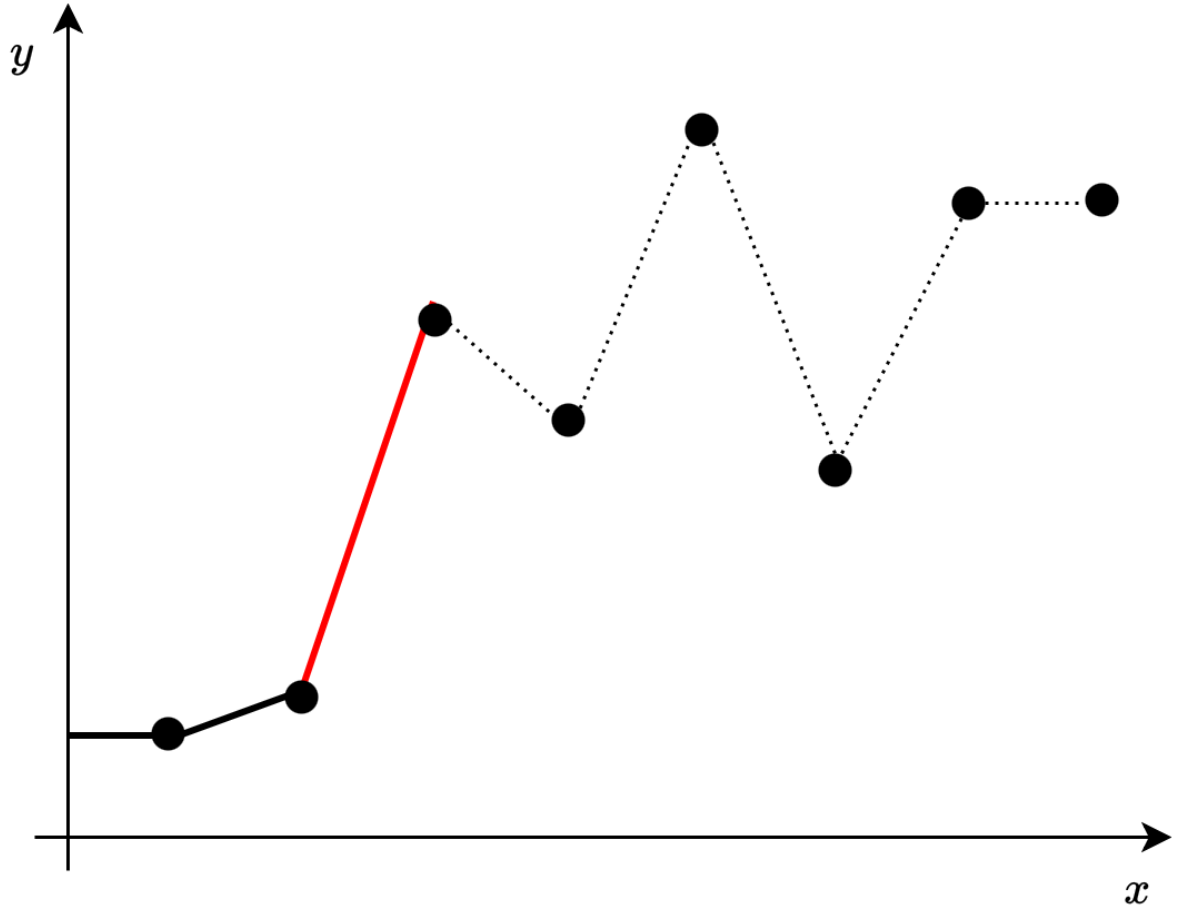
- 帰着後の問題の復習：
  - M 個の点 $(1, Y_1), (2, Y_2), \ldots, (M, Y_M)$ がある
  - 各クエリでは L_q, R_q, A_q が与えられるので、点 $[L_q, R_q)$ のうち、傾き A_q の直線を右から動かした時に最初にぶつかる点を知りたい
  - 本小課題では、常に $L_q = 1, R_q = M + 1$（全ての点を考慮）

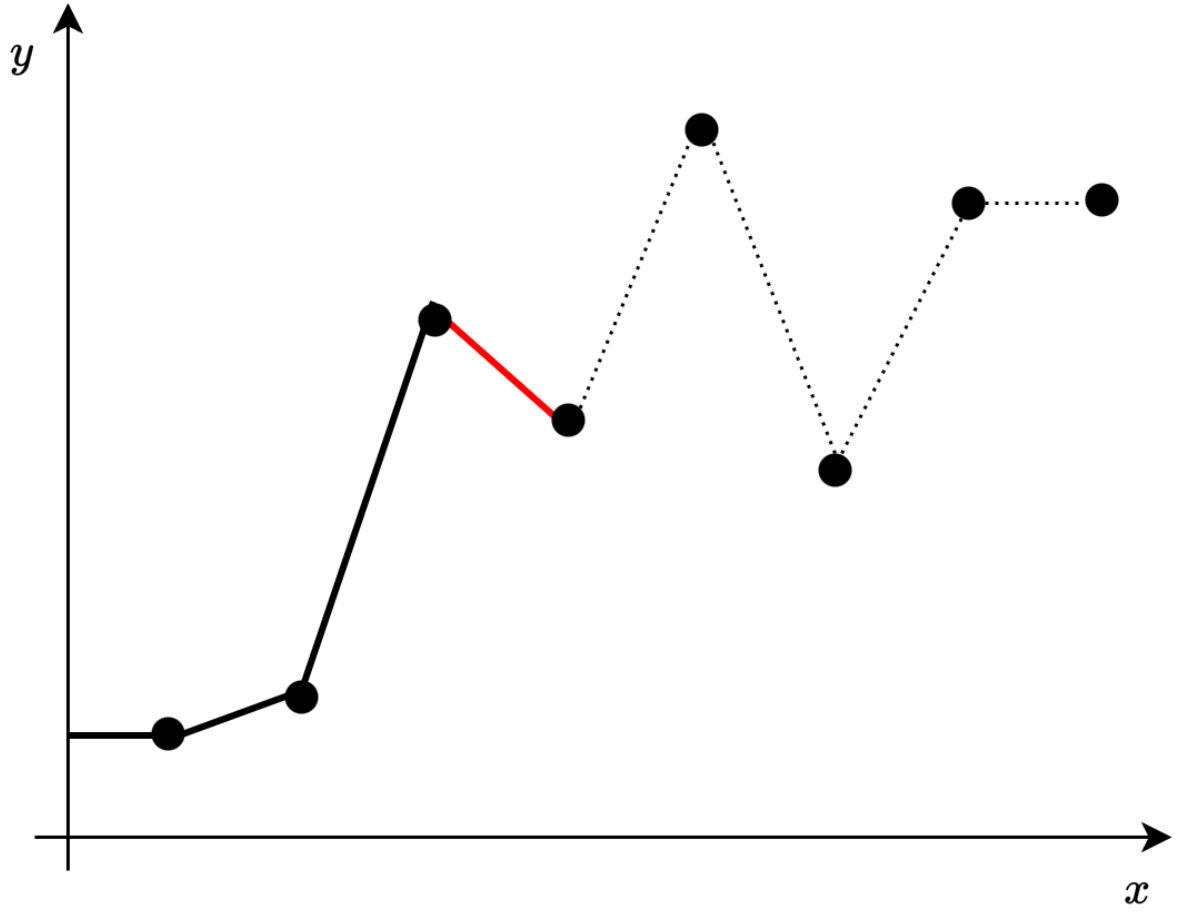


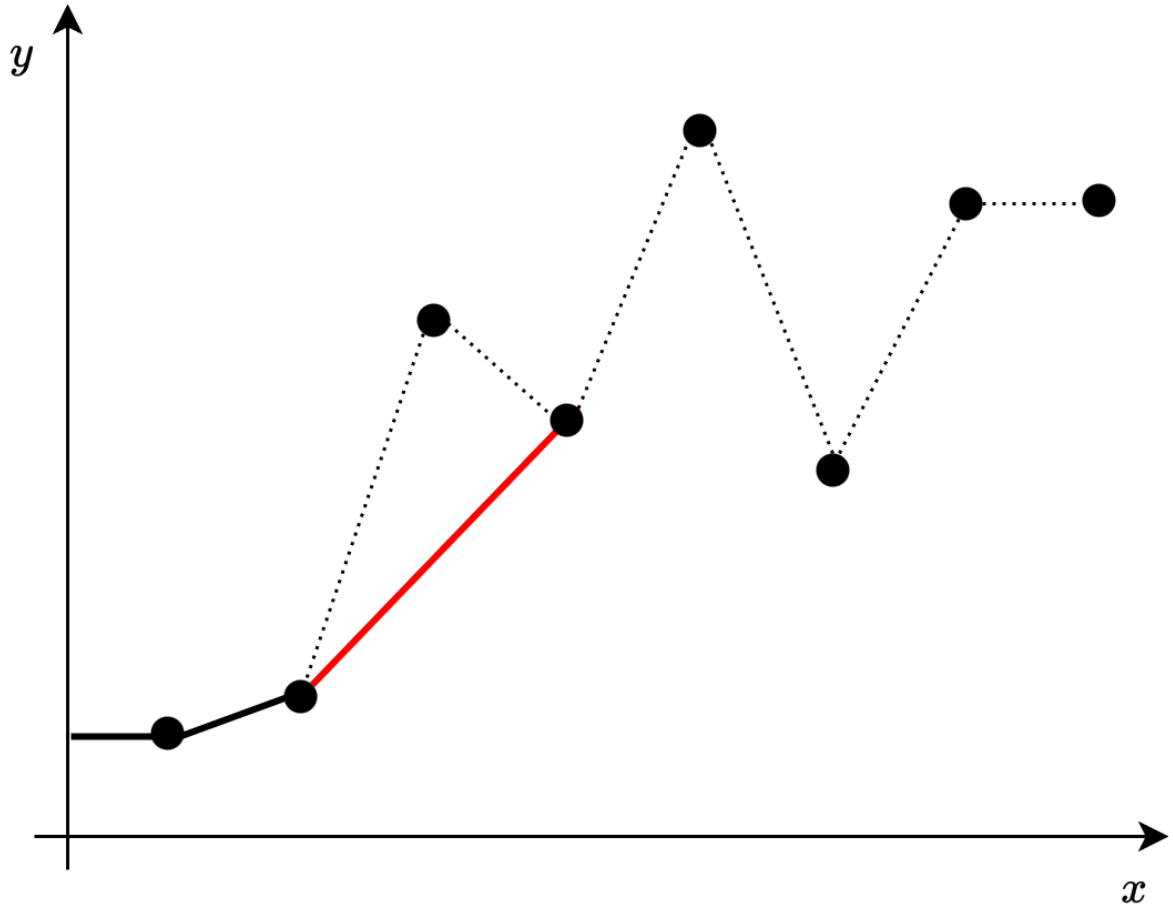

y
x

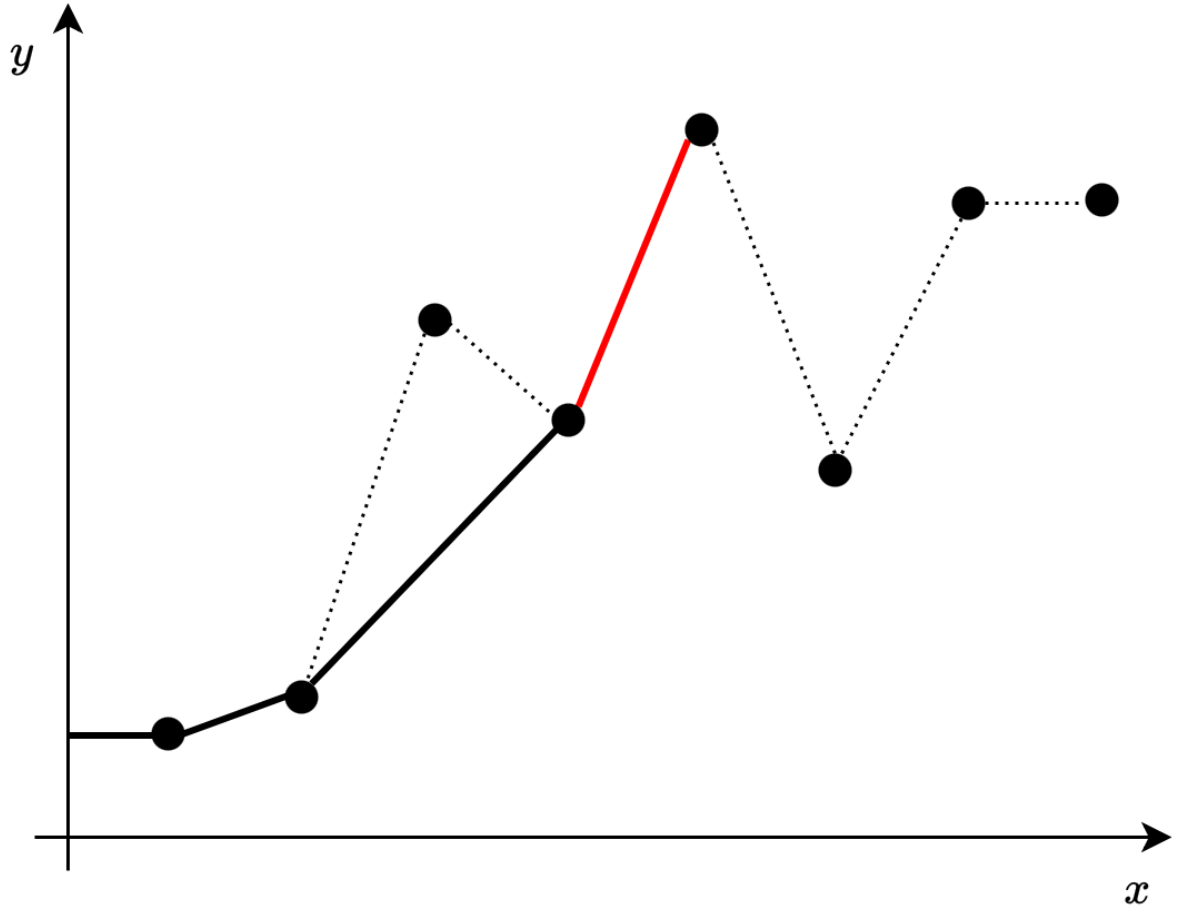

y
x

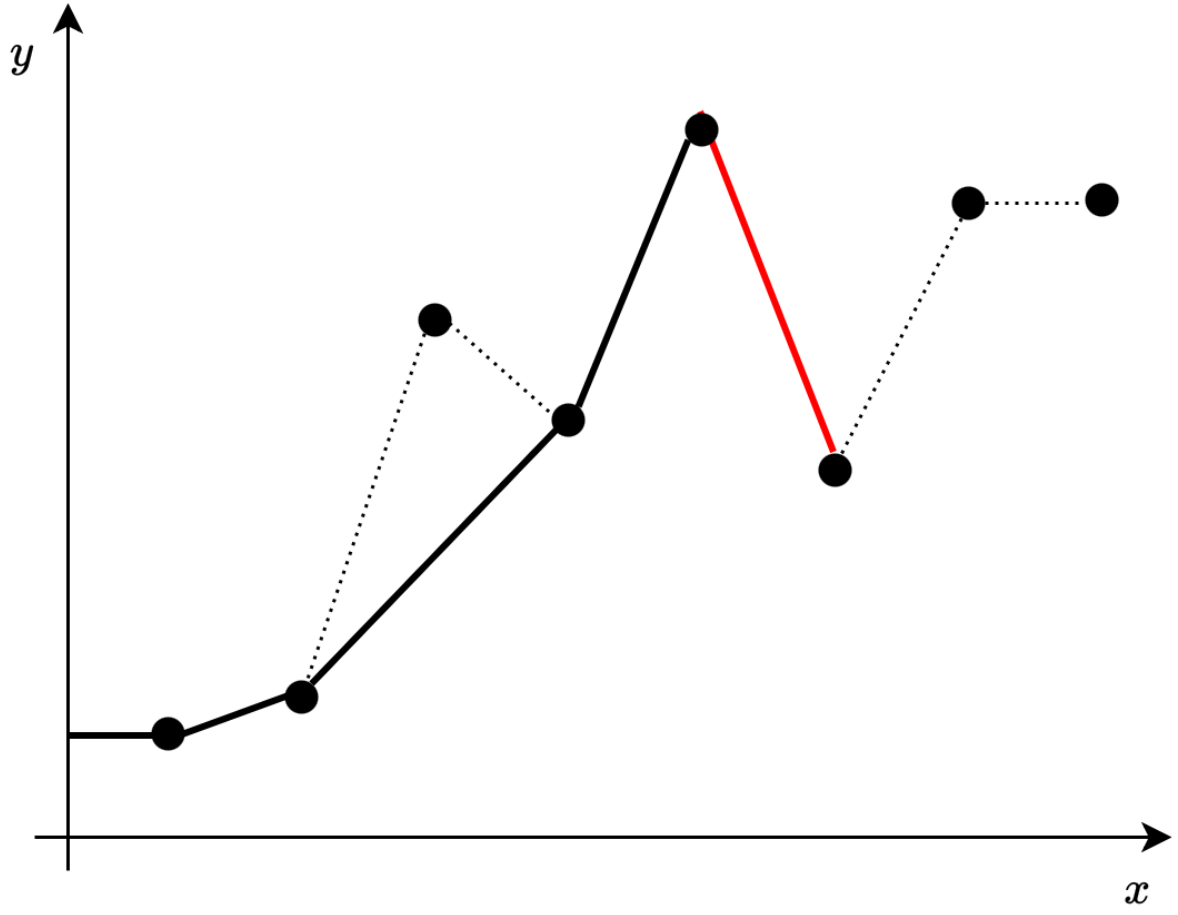

y
x

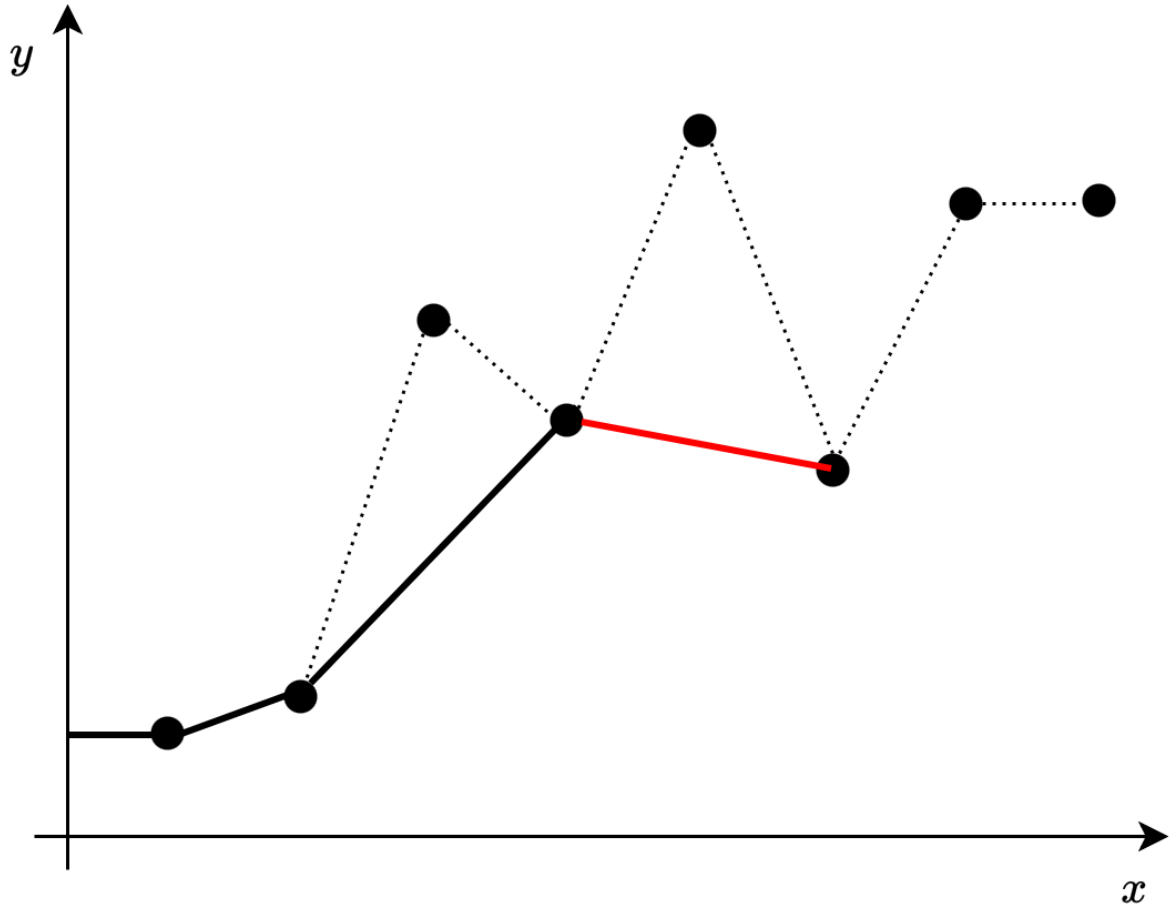

y
x

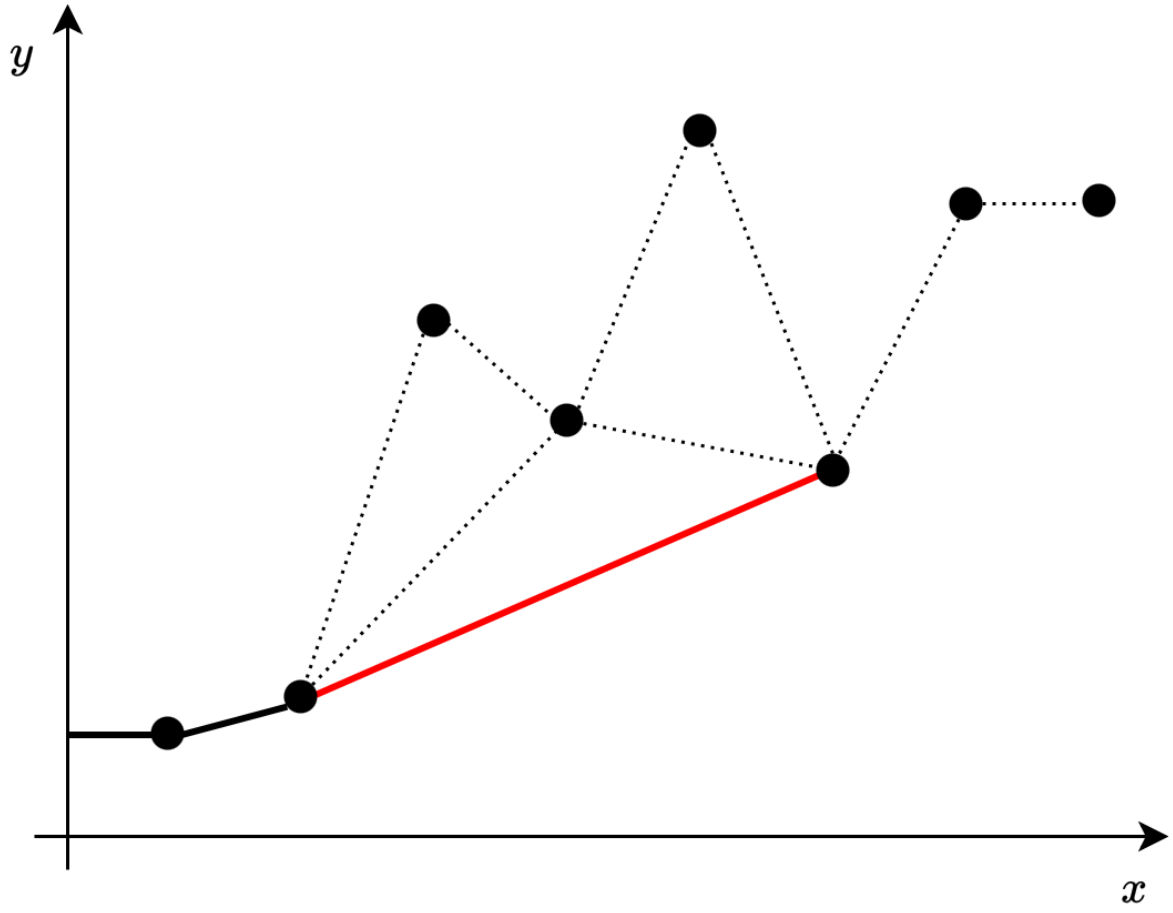

y
x

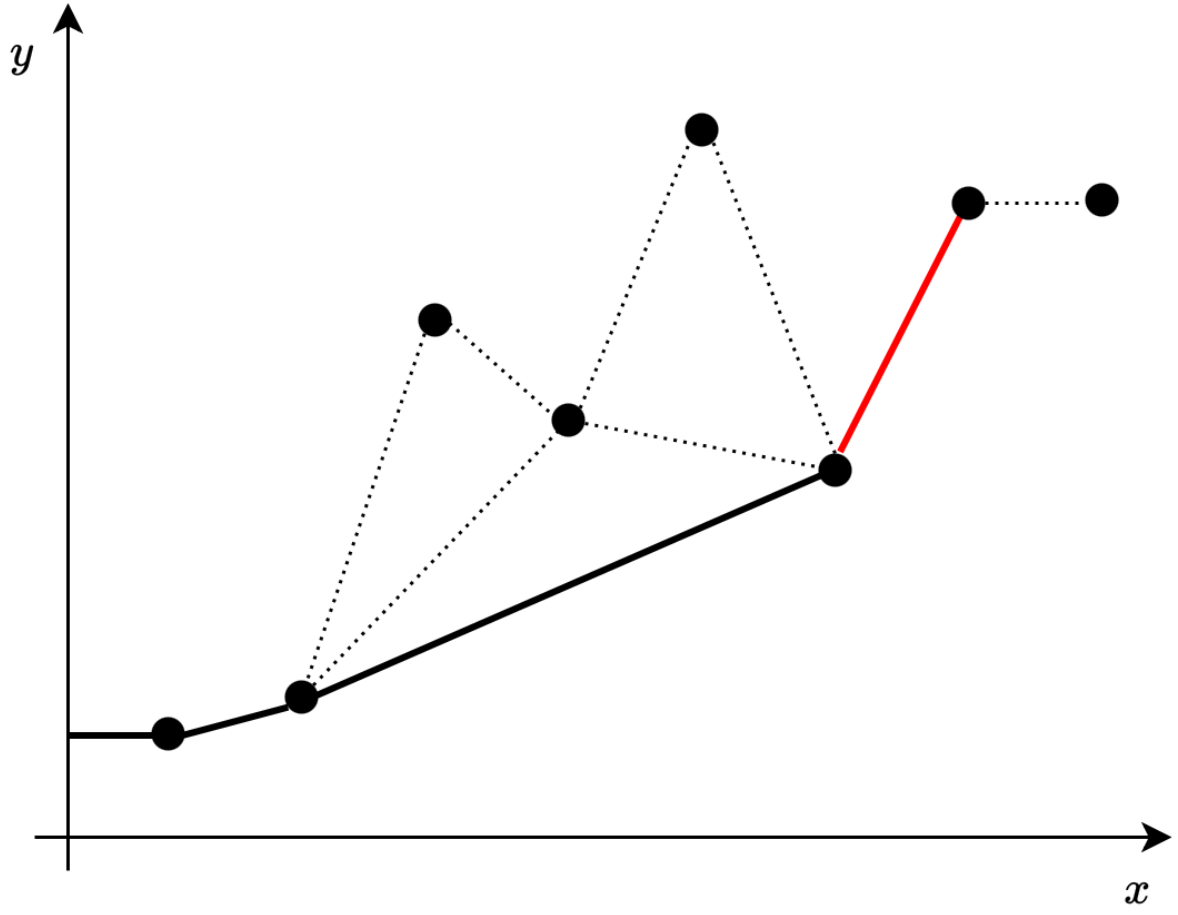

y
x

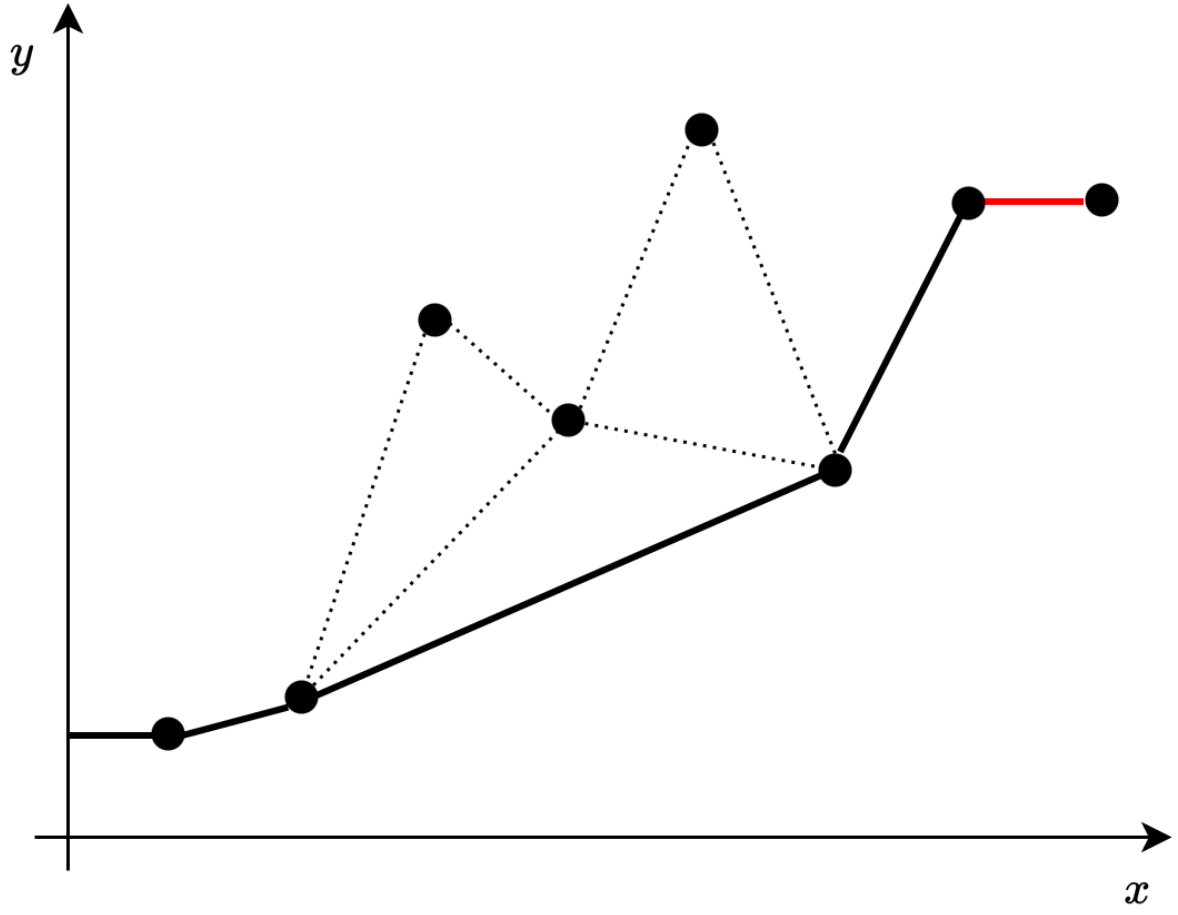

y
x

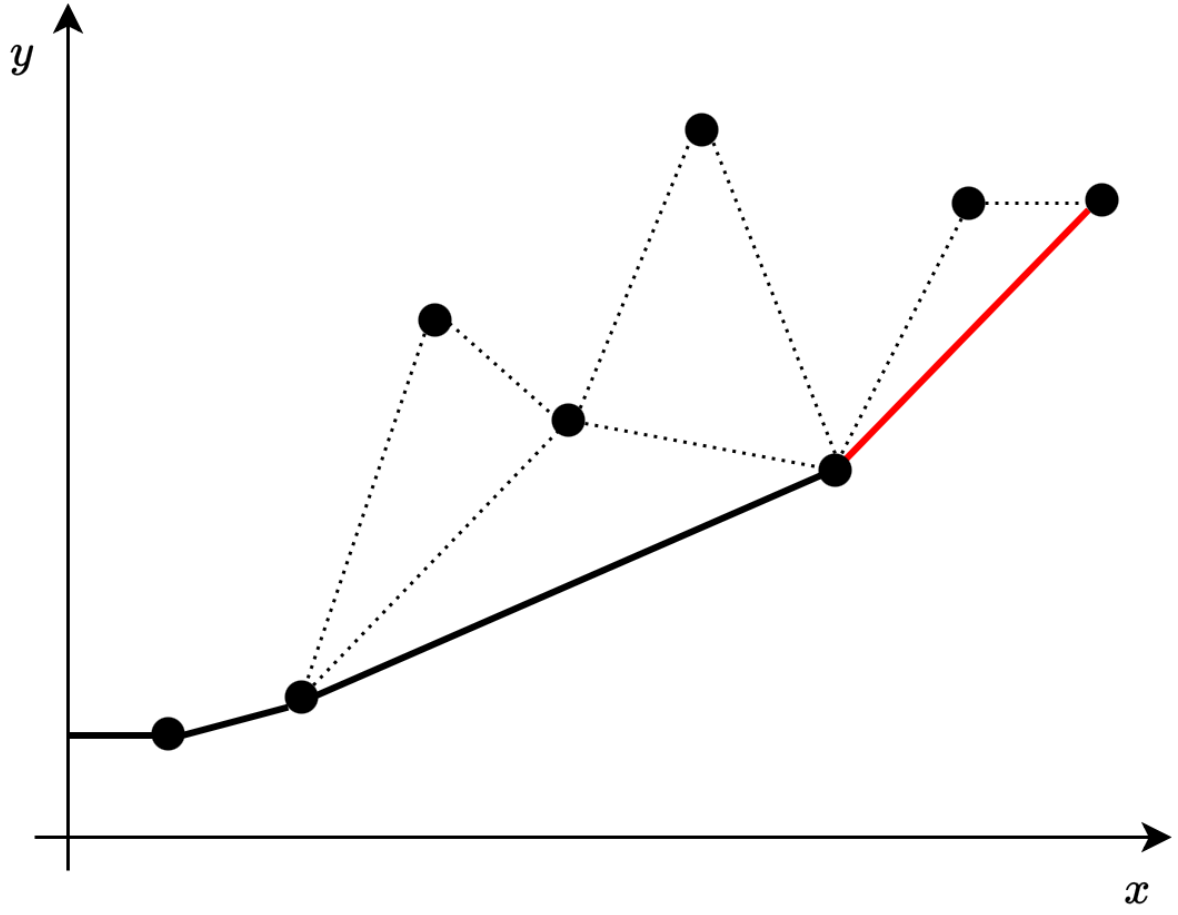

y
x

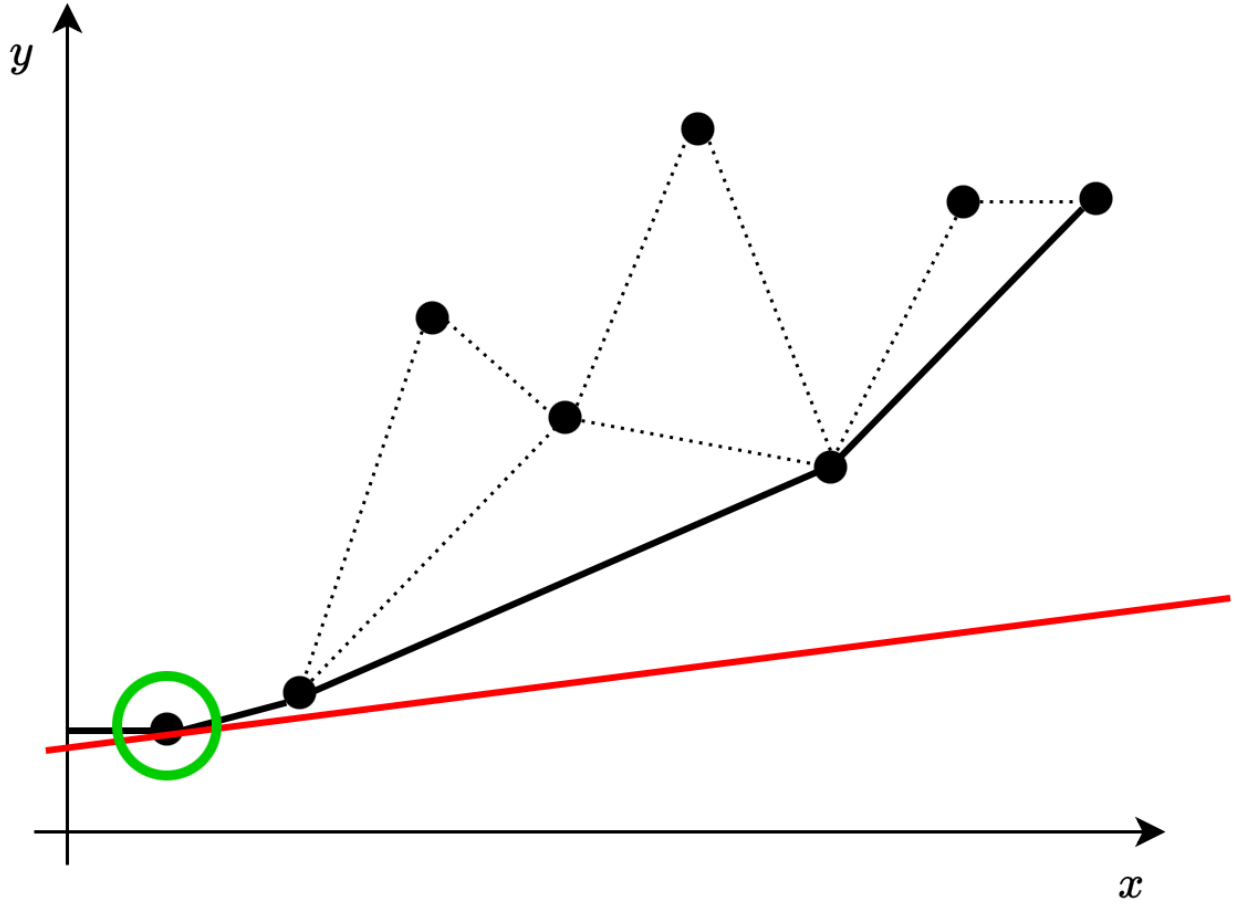

y
x

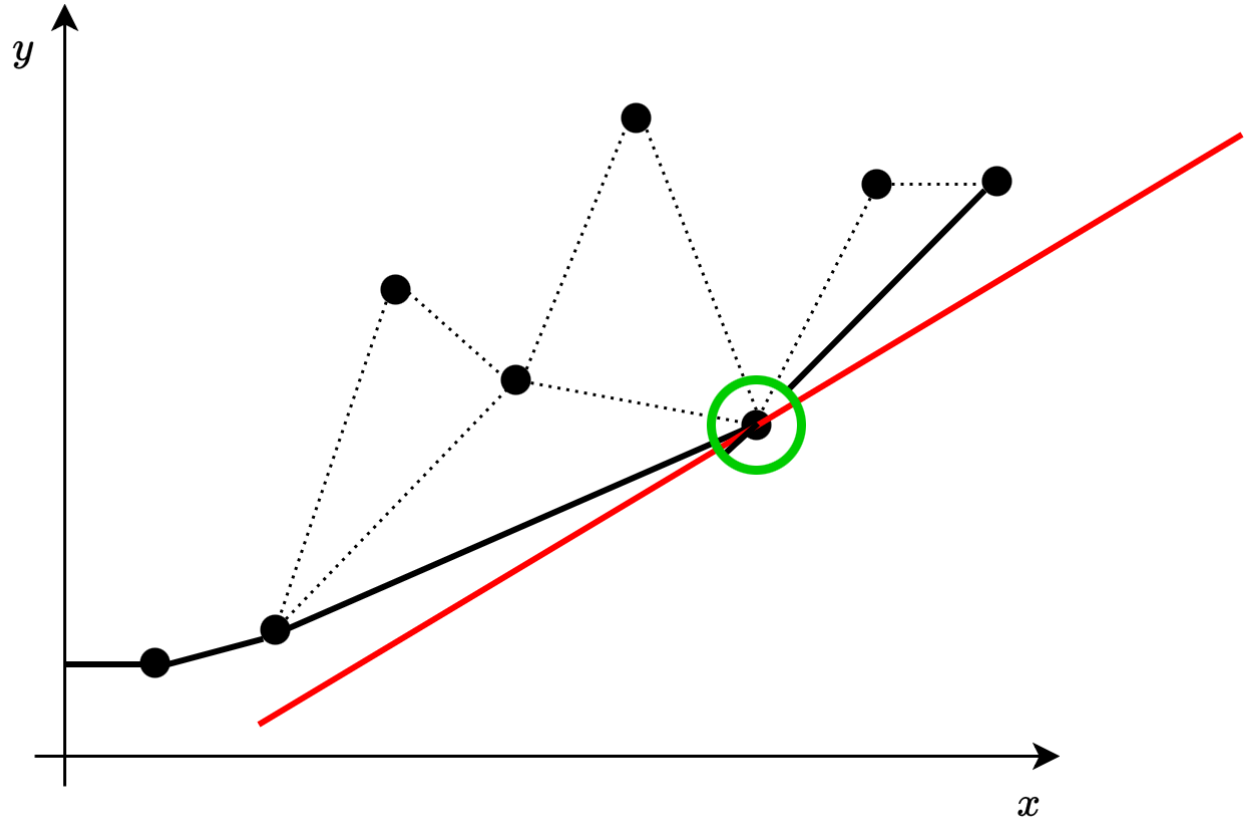

y
x


y
x


y
x


y
x


y
x


y
x

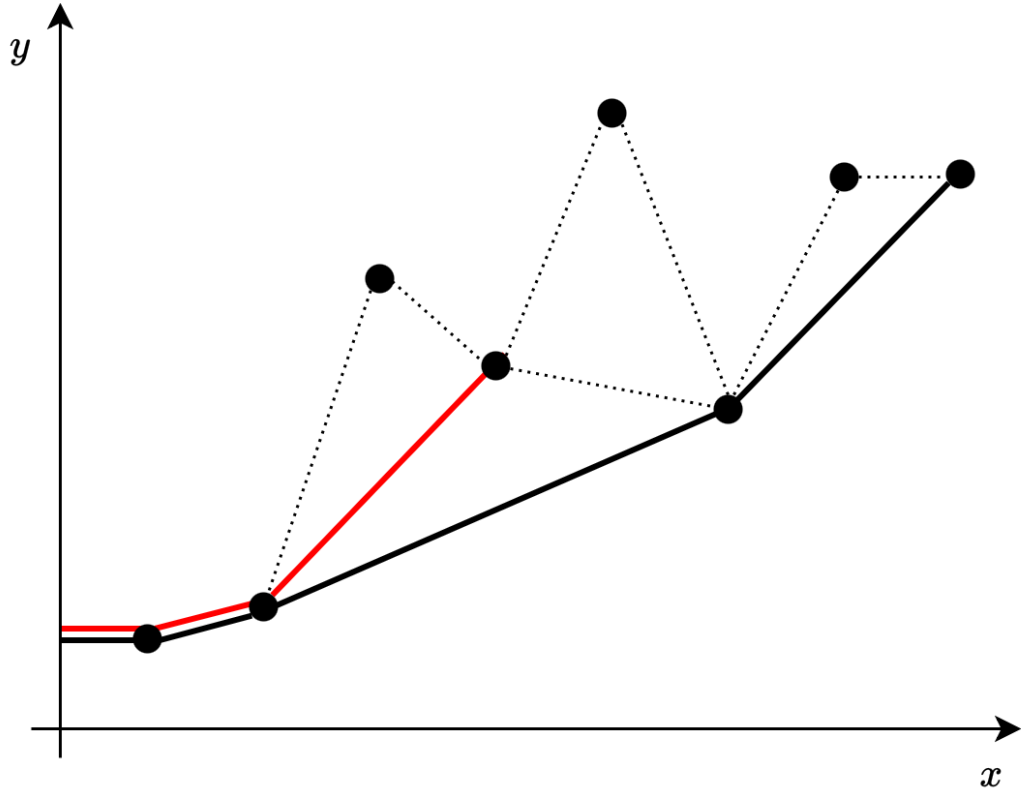
Subtask 3 : $T_M \leq B_q < T_1 + L$

- 与えられた M 個の点の（下側）凸包を前計算（元々ソートされているので線形時間で可能）
- 各クエリについて、
  - 傾き A_q の直線を右から動かした時に最初にぶつかる点を知りたい（復習）
  - 凸包に含まれる点のみを考慮すればよい（証明略）
  - 凸包を構成する線分（傾きの昇順に並んでいる）について、

    A_q 未満から以上に変化する場所を二分探索で見つける
  - クエリあたり $O(\log M)$ 時間

Subtask 4 : $T_M \leq B_q$

- 先程との違い：L_q が任意の値を取るようになった（$R_q = M + 1$ は保たれたまま）
- 点 $L_q, L_q + 1, \ldots, M$ のみを考慮した凸包上で二分探索をしたい
  - ~~全体の凸包から該当範囲内の点を取り出すだけ？~~



全体の凸包

点 1 - 4 の凸包

Subtask 4 : $T_M \leq B_q$

- 先程との違い：L_q が任意の値を取るようになった（$R_q = M + 1$ は保たれたまま）
- 点 $L_q, L_q + 1, \ldots, M$ のみを考慮した凸包上で二分探索をしたい
  - ~~全体の凸包から該当範囲内の点を取り出すだけ？~~
- 点 $M, M - 1, \ldots, 1$ の順に点を追加していきながら凸包を構築することを考える
  - 先程のアルゴリズムの逆バージョン
  - 全ての l について、点 $l, l + 1, \ldots, M$ のみを考慮した凸包が途中で登場！
- クエリを L_q の昇順にソートして、全体 $O(M + Q(\log Q + logM))$ 時間

Subtask 5 : $M \leq 5\times10^5, Q \leq 10^5$

- L_q も R_q も固定されなくなってしまった！
  - 点 $L_q, L_q+1, \ldots, R_q$ における $-A_q x + y$ の最小値を求めたい
  - 区間クエリなので、Segment Tree のようなことをしてみたい
- Segment Tree の各ノードについて、その区間内の点からなる凸包を前計算
- 各クエリでは、$[L_q,\ R_q)$ を $O(\log M)$ 個の区間に分解し、

  各区間において凸包上の二分探索で $-A_q x + y$ の最小値を求める
- 全体 $O(Mlog\ M + Qlog^2 M)$ 時間

Subtask 6：$M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- 複数の解法があるが、ここではそのうちの一つを紹介（他の例：分割統治）
- 先程の Segment Tree 解法では、任意の区間に対するクエリを処理できた
  - 実際には L_q, R_q は何でも好きに選べるわけではない
  - $L_q = min\, i$ s.t. $T_i \geq B_q - L$； $R_q = max\, i$ s.t. $T_i \leq B_q$；
  - クエリを B_q の昇順にソートすると、**L_q と R_q も昇順にソートされる**

Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- 以下、$L_1 \leq L_2 \leq \ \ldots \leq L_Q,\ R_1 \leq R_2 \leq \ \ldots \leq R_Q$ を仮定
- 与えられた点列の連続部分列 $[L_1, R_1], [L_2, R_2], \ldots, [L_Q, R_Q]$ それぞれについて凸包を知りたい
- やりたいこと：空の点列から始めて、
  - 末尾への点の追加
  - 先頭からの点の削除
  - 現在の点列における凸包へのアクセス（具体的には、二分探索）
- Queue のような操作

Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- ところで、以下は実現可能だろうか？
- 空の点列から始めて、
  - 末尾への点の追加
  - 末尾からの点の削除
  - 現在の点列における凸包へのアクセス

Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- ところで、以下は実現可能だろうか？
- 空の点列から始めて、
    - 末尾への点の追加
    - 末尾からの点の削除
    - 現在の点列における凸包へのアクセス
- できる（各点を追加する際に、その追加によって削除された点を記録しておく）
- Stack のような操作

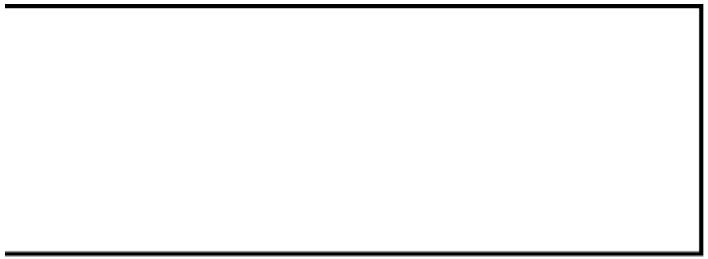
Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

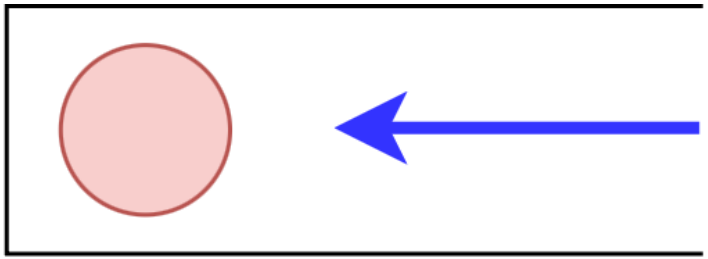
- やりたいのは Queue のような操作（末尾追加・先頭削除・凸包アクセス）
- Stack のような操作はできる（末尾追加・末尾削除・凸包アクセス）
- つまり……?

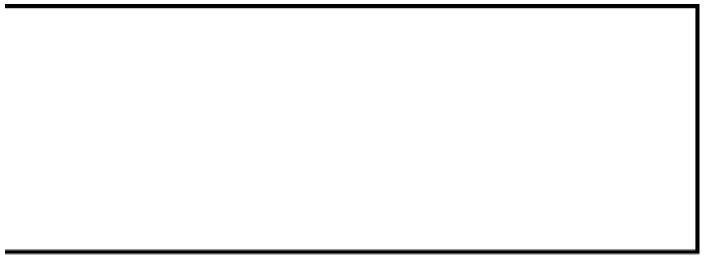
Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

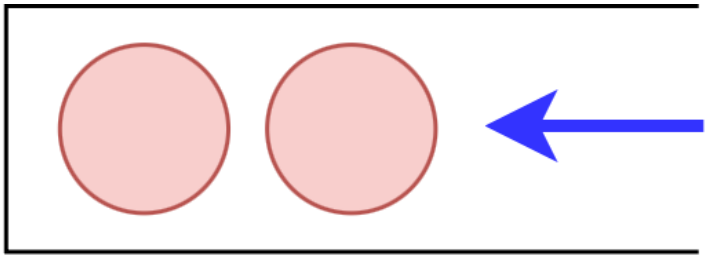
- やりたいのは Queue のような操作（末尾追加・先頭削除・凸包アクセス）
- Stack のような操作はできる（末尾追加・末尾削除・凸包アクセス）
- つまり……?

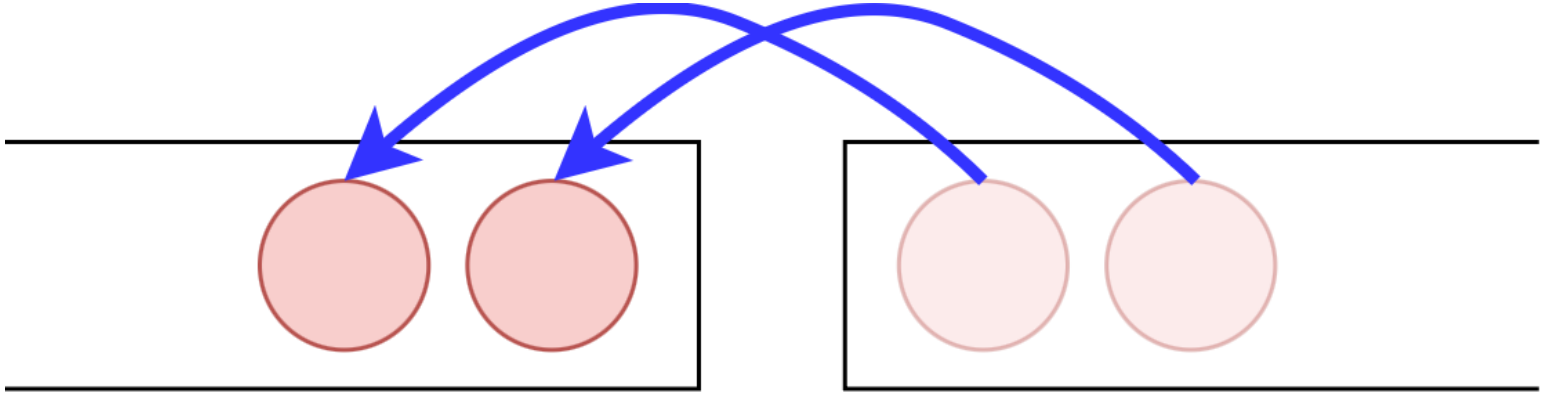
Two Stacks Queue

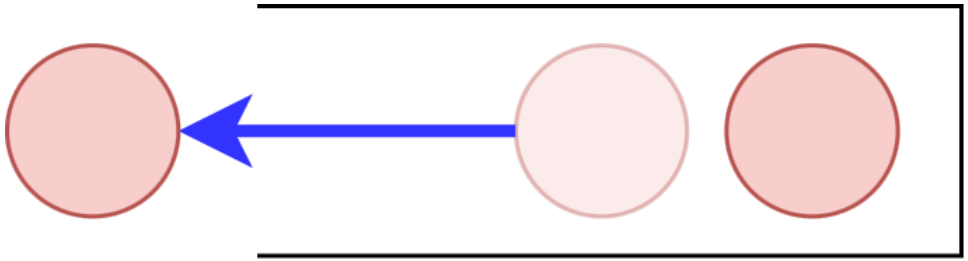


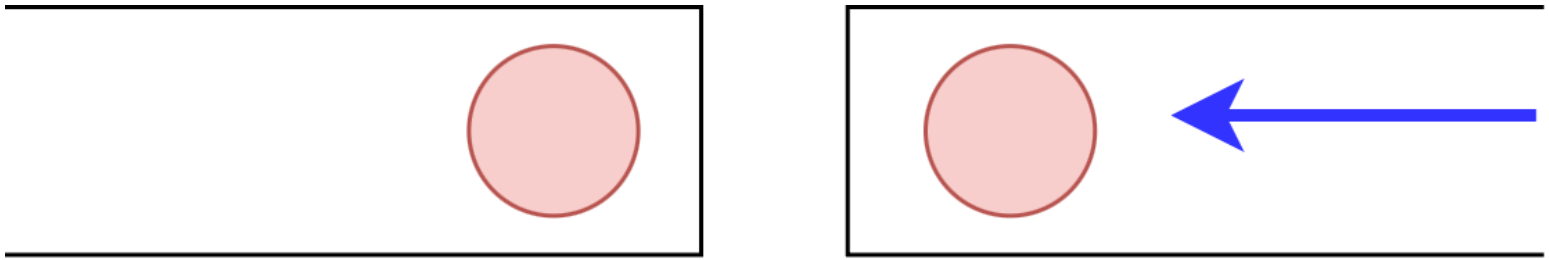












Subtask 6 : $M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- Two Stacks Queue
  - 2 つのスタックを背中合わせにくっつけて Queue と同等の操作を実現
  - 末尾追加：末尾用スタックに追加
  - 先頭削除：先頭用スタックが空でないならそこから削除

    空なら、一旦末尾用スタックの中身を全て先頭用に移動する
  - 各要素は高々 1 回しか移動されないので、ならし $O(1)$

Subtask 6：$M \leq 2\times10^6, Q \leq 4\times10^5$ (満点)

- これを応用して、各スタックにおいて点列とそれらの凸包を両方管理すれば OK
  - クエリ処理：両スタックの凸包それぞれに対して二分探索
  - 移動処理（先頭スタックが空のときの先頭削除）：

    末尾用スタックに含まれる元の点列を参照して、線形時間で一から凸包構築

    （末尾用スタックの凸包をそのままコピーするだけだと先頭削除ができない！）
- 全体 $O(M + Q(logQ + logM))$ 時間

得点分布

得点	小課題	人数
100	1, 2, 3, 4, 5, 6	0
82	1, 2, 3, 4, 5	1
72	1, 2, 3, 5	1
60	1, 2, 3, 4	3
50	1, 2, 3	5
42	1, 2, 5	1
20	1, 2	15
0	ø	4