Convex Hull Trick

CHT আসলে একটা Data Structure এর মত, যেইটা কিছু Linear Function $f_i(x) = m_i x + b_i$ maintain করে। আর আমরা Query করতে পারি কোন একটা x এর জন্য $max/min\{f_i(x)\}$ কত হতে পারে।

CHT আসলে একটা Data Structure এর মত, যেইটা কিছু Linear Function $f_i(x) = m_i x + b_i$ maintain করে। আর আমরা Query করতে পারি কোন একটা x এর জন্য $max/min\{f_i(x)\}$ কত হতে পারে।
যেমনঃ CHT তে যদি এই function গুলা থাকে -

- $f_1(x) = 4x + 3$
- $f_2(x) = 3x - 2$
- $f_3(x) = -5x + 15$

তাহলে আমরা 4 দিয়ে Query করে Maximum চেলে উত্তর পাব $max\{f_1(4), f_2(4), f_3(4)\} = max\{19, 10, -5\} = 19$

Convex Hull Trick এর ২টা Version আছে -

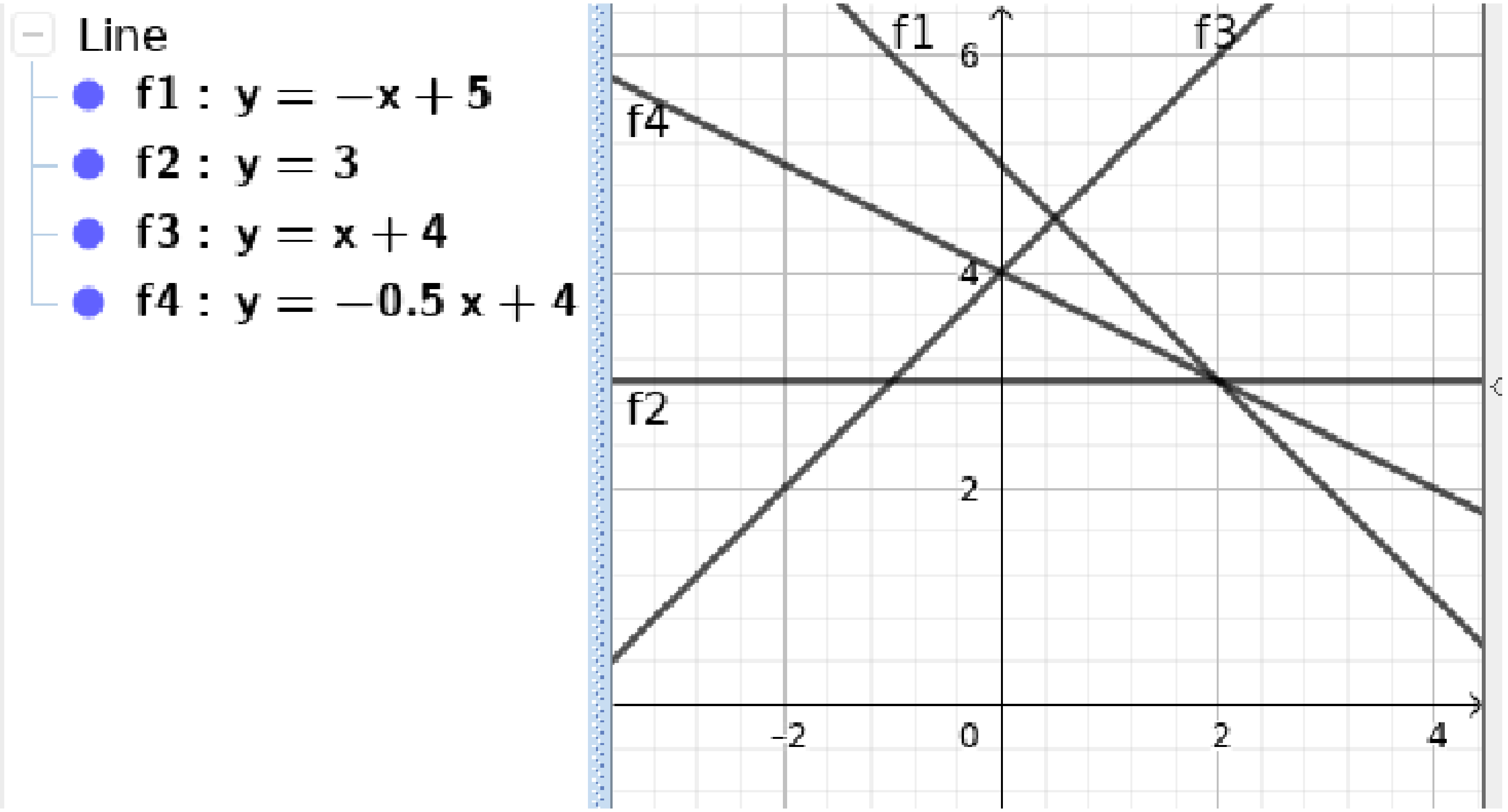
- এক ধরনের CHT তে আমরা ইচ্ছা করলেই Random line add করতে পারি না, কিছু Condition মানতে হয়, যেমনঃ $m_1 \geq m_2 \geq \cdots \geq m_i \geq m_{i+1}$ অথবা $m_1 \leq m_2 \leq \cdots \leq m_i \leq m_{i+1}$ এইরকম। এটাকে আমরা Semi-offline CHT বলতে পারি।
  এইধরনের CHT তে আমরা $O(1)$ Amortized (অর্থাৎ $O(n)$ overall) Complexity তে নতুন Line add করতে পারি। আবার $O(\log n)$, অথবা বিশেষ শর্ত সাপেক্ষে $O(1)$ Amortized Complexity তে Query করতে পারি।
- আরেক ধরনের আছে যেইটাতে ইচ্ছা মত Random Line Add করা যায়। সেইটাকে Dynamic CHT বলে। এইধরনের CHT তে $O(\log n)$ এ নতুন Line add বা Query করা যায়।

প্রথমে আমরা যা করব, Function গুলাকে Plot করে ফেলব।

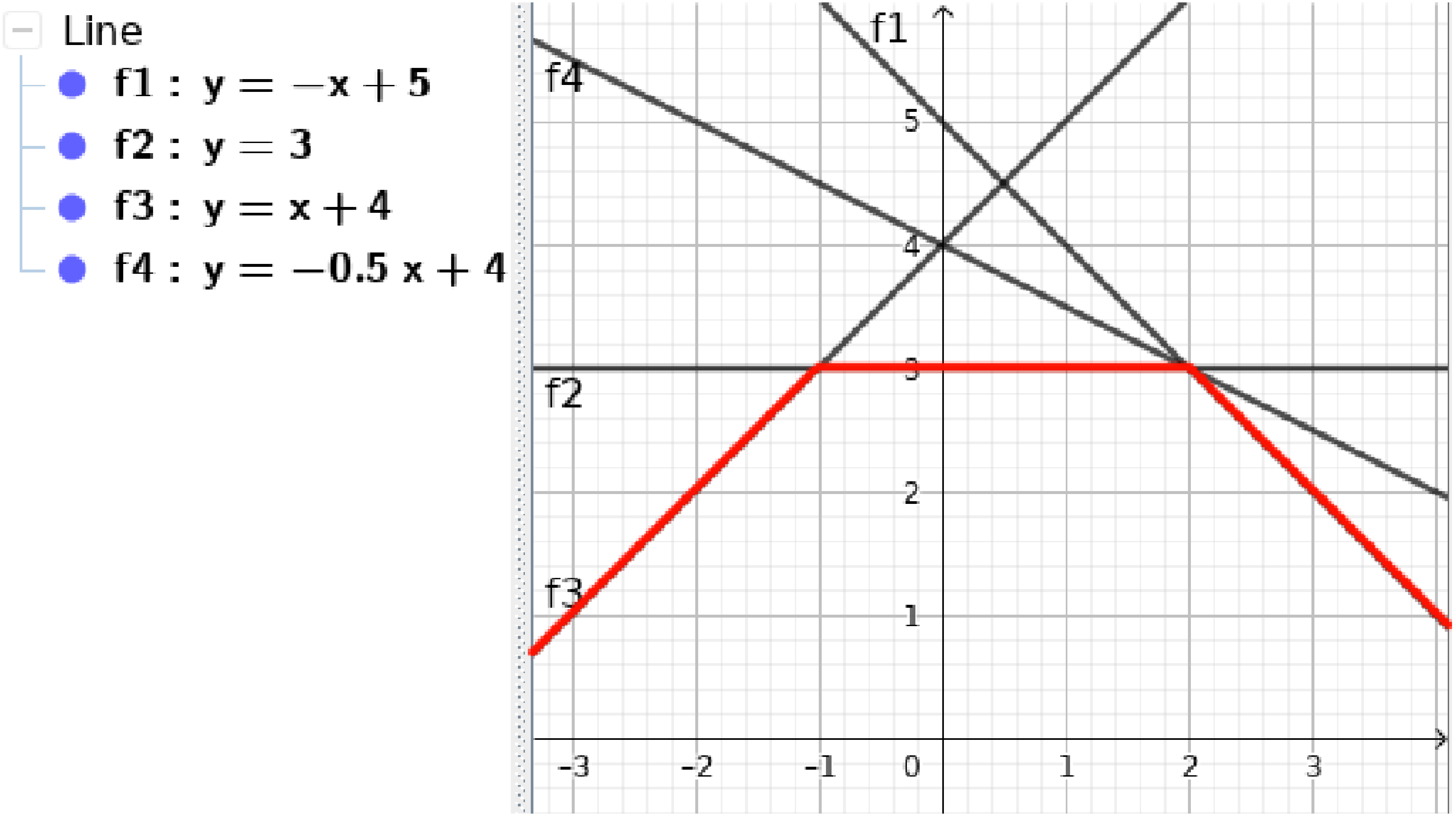
মনে করি, আমাদের Function গুলা হল -

$$f_1(x) = -x + 5$$
$$f_2(x) = 5$$
$$f_3(x) = x + 4$$
$$f_4(x) = \frac{-1}{2}x + 4$$

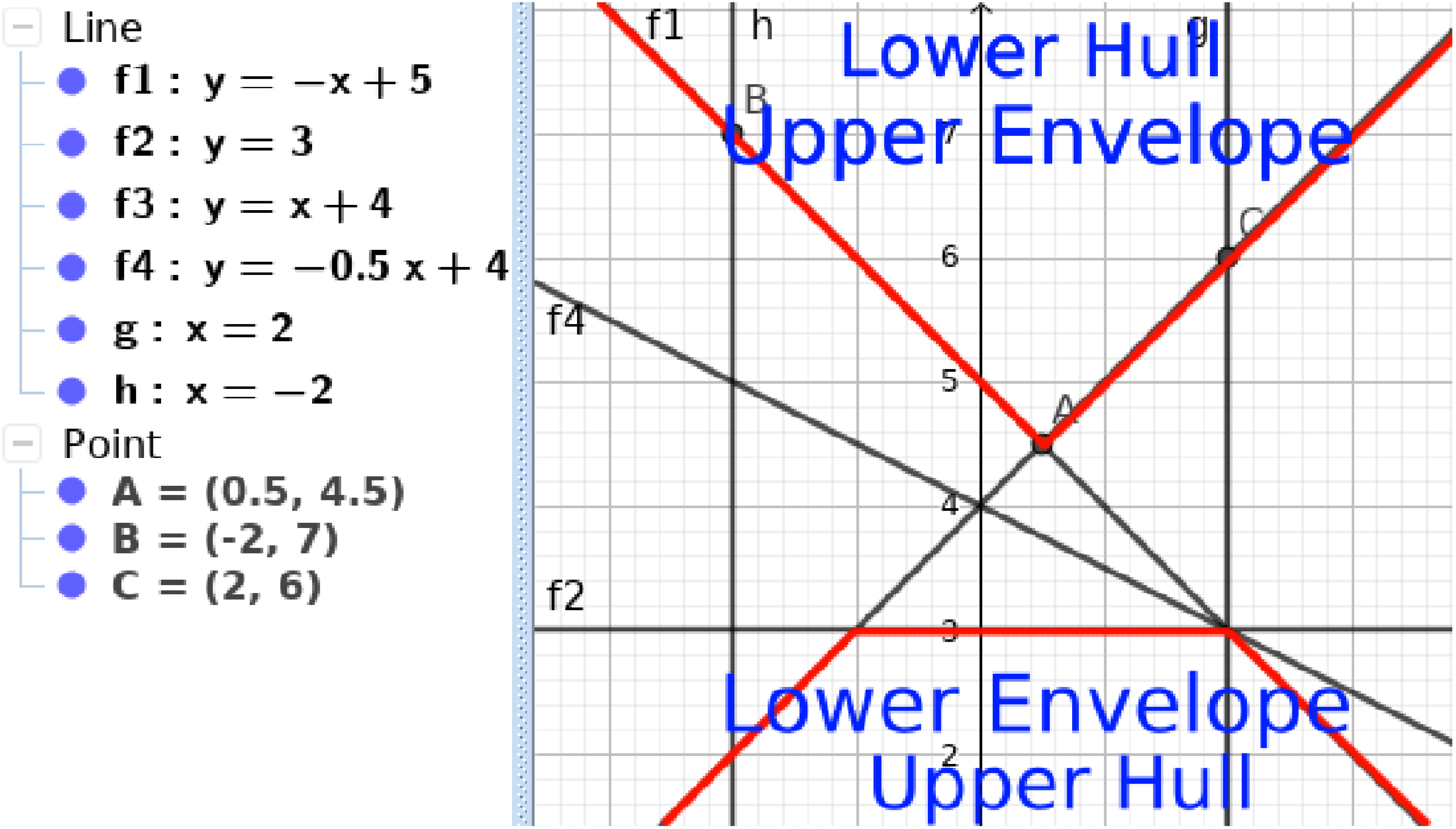
Function গুলা Plot করে ফেলি -



আরেকটা জিনিস আমরা লক্ষ্য করতে পারি, যেই Line গুলা আমরা CHT তে রাখতে চাচ্ছি, তাদের যেই অংশ গুলা কাজে লাগে সেইগুলা একটা Half Convex Hull গঠন করে। এদের Upper Hull/Lower Envelope এবং Lower Hull/Upper Envelope বলে। যেমন উপরের Function গুলার জন্য যদি আমরা Minimum maintain করতে চাই তাহলে শুধু এই অংশ গুলা কাজে লাগে -



আবার একই ভাবে আমরা যদি Maximum maintain করতে চাই তাহলে Lower Hull/Upper Envelope maintain করলেই হবে -



Consider the following problem. There are n cities. You want to travel from city 1 to city n by car. To do this you have to buy some gasoline. It is known that a liter of gasoline costs $cost_k$ in the k^{th} city. Initially your fuel tank is empty and you spend one liter of gasoline per kilometer. Cities are located on the same line in ascending order with k^{th} city having coordinate x_k. Also you have to pay $toll_k$ to enter k^{th} city. Your task is to make the trip with minimum possible cost. It's obvious that the solution can be calculated via dynamic programming:

$$dp_i = toll_i + \min_{j<i}(cost_j \cdot (x_i - x_j) + dp_j)$$