⇒ Get all edges. (unique & sort)

eg: (2, 9, 10)   (3, 7, 15)   (5, 12, 12) (5, 20, 10)
                                           (19, 24, 8)

⇒ so ⎡ 2 , 3 , 5 , 7 , 9 , 12, 15, 19, 20,
      ↓   ↓   ↓   ↓   ↓   ↓   ↓   ↓   ↓
      0   1   2   3   4   5   6   7   8

                                         24
                                          ↓
                                          9

      ↳ map ⎡ 2 → 0   24 → 9 ⎤
            ⎢ 3 → 1   ⎥
            ⎣ - - -   ⎦

⇒

       15 15
      ┌────┬────┬────┬────┬────┬────┬────┬────┬────┬────┐
      │ 10 │ 10 │ 10 │ 10 │   │   │   │   │   │   │
      └────┴────┴────┴────┴────┴────┴────┴────┴────┴────┘
height 0   1   2   3   4   5   6   7   8   9

iterate through buildings.
find building [0], building [1]

Final

      ┌────┬────┬────┬────┬────┬────┬────┬────┬────┬────┐
  ↳   │ 10 │ 10 │ 10 │ 10 │ 12 │ 0 │ 10 │ 10 │ 8 │ 0 │
      └────┴────┴────┴────┴────┴────┴────┴────┴────┴────┘
        0   1   2   3   4   5   6   7   8   9
         15   15   12

ans ⇒ (2, 10)   (3, 15) (7, 12) (12, 0)

              (15, 10) (20, 8) (24, 0)

      position height.

2nd approach (Sweep Line)

x_1 x_2 x_3 x_4 x_5 x_6

Sweep a line vertically at the edges, find the maximum height.

Sweep Line + Priority Queue
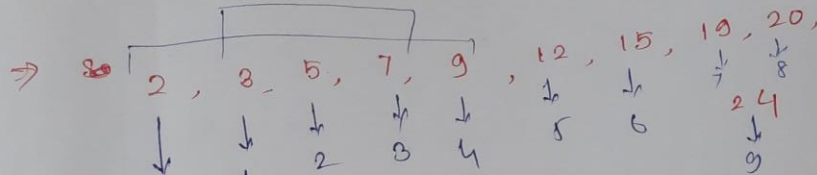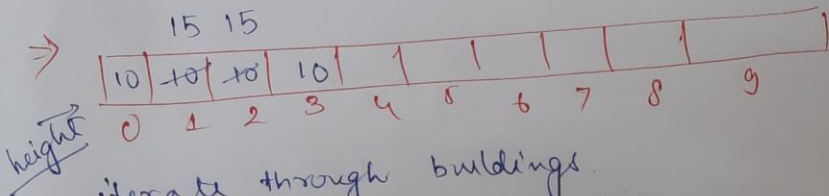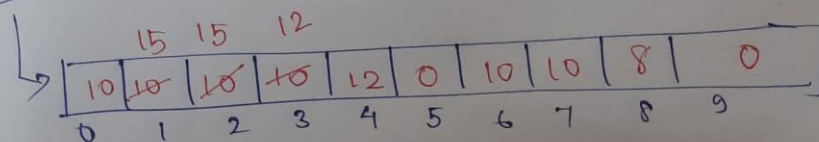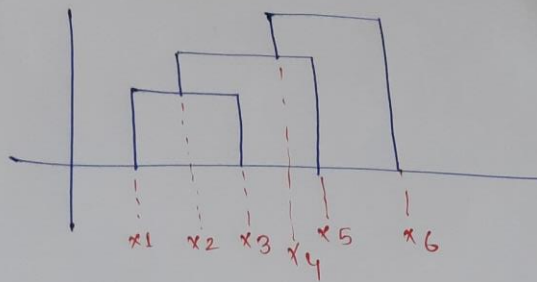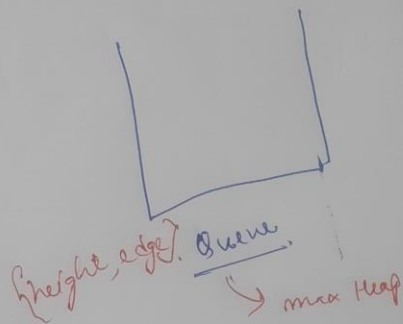
(2, 9, 10)   (3, 7, 15)   (5, 12, 12)   (15, 20, 10)
  0   1   2   3

                                            (19, 24, 8)
                                              4

2 → 0
3 → 1
5 → 2
7 → 1
9 → 0
12 → 2
15 → 3
19 → 4
20 → 3
24 → 4

{height, edge}. Queue.
                    → max Heap

Push {height, edge}