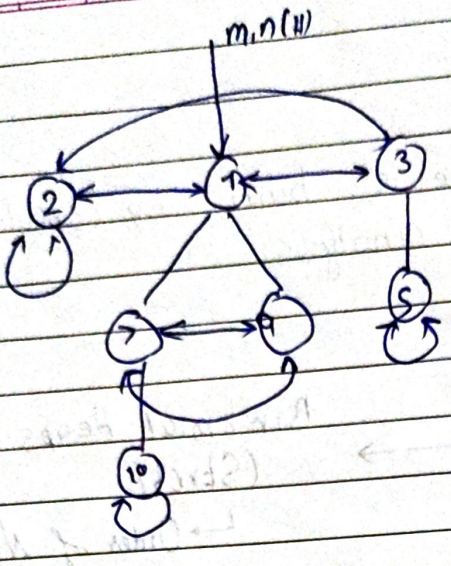
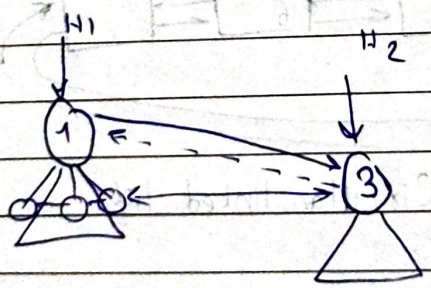
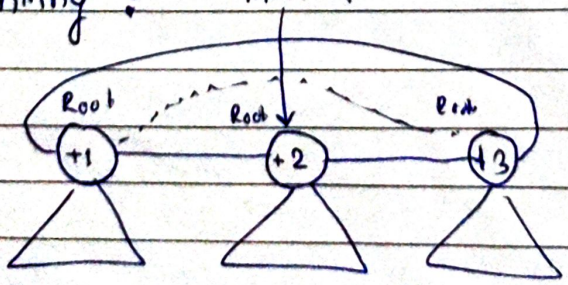


min(H) means pointer to root element with lowest value.

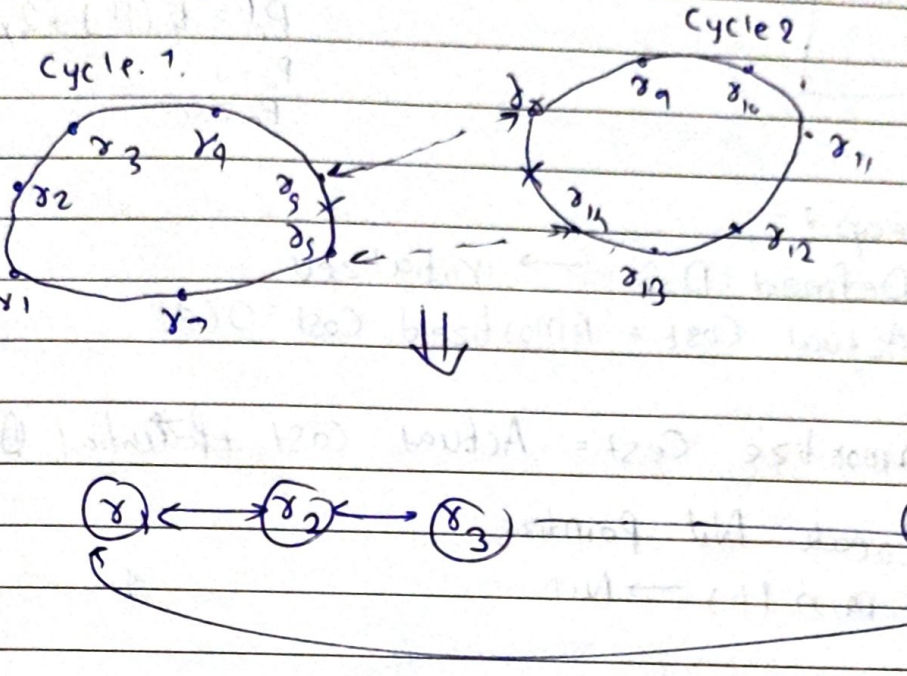
* Linking: O(1)
Given two order tree, make root with bigger key the child other root.



* Unlinking : min(H)



* Merging of Cycles.



The fibonacci heap data structure utilizes following potential function

- let $Z_1, Z_2, \ldots Z_n$ be the Sequence of operations. (Any sequence)

- let $t_i(H)$ = No of roots in the root list (i.e doubly linked list of all roots) after the operation Z_i

$m_i(H)$ = Number of marked nodes in H after Z_i Operation.
[* Some Nodes are Marked]

* Potential Function: (P_i)

$P_i = t_i(H) + 2 \cdot m_i(H)$

P_i - Potential of D.S after i^{th} Operation

$PD = P_i - P_{i-1} = t_i(H) + 2 m_i(H) - t_{i-1}(H) - 2 m_{i-1}(H)$

Fibonacci Sequence $F_0, F_1, F_2, \ldots$ $F_n = F_{n-1} + F_{n-2}$ ⟷ related with Golden Ratio.

Initially
$P_0 = 0$
$P_n \geq 0$

Potential Function
$P_i = t_i(H) + 2m_i(H)$
$P_0 = 0$
$P_n \geq 0$.

1. Make_heap :
   Defined D.S ⟶ initialized.
   Actual Cost = Amortized Cost = O(1)

   Amortize Cost = Actual cost + Potential Diff.

   Create Nil Pointers
   ~~min (H)~~ ⟶ Nil.

2. Minimum (H) : —
   It returns the minimum heap.
   Minimum(H) will not affect No of Nodes & No of Mark nodes.

3. Meld (H_1, H_2) ⟷
   (Meld or Union) Union (H_1, H_2).
   Steps : - 1] Merge the Root Cycle of H_1 and H_2 — O(1)
   2] Adjust Min (H) to be minimum (min(H), min (H_2)) — O(1).

   ∴ Actual Cost = O(1) + O(1) = O(1)

   Potential Change.

4. Insert (H, x
   Steps -
   1. Create Fibonacci Heap F-heap H, which has only one node x.

2. Meld $(H, H_1) \rightarrow H'$ (New Heap)

$P_i(H') = t_{i-1}(H) + 1 + 2\, m_{i-1}(H)$

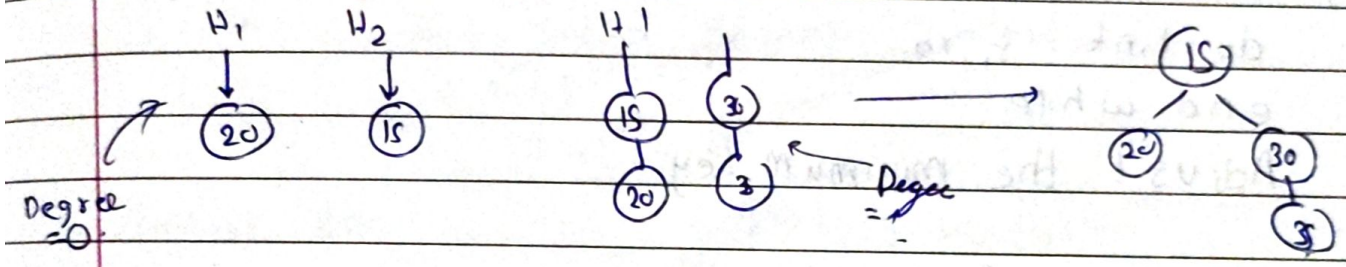
$=$

(New Node)

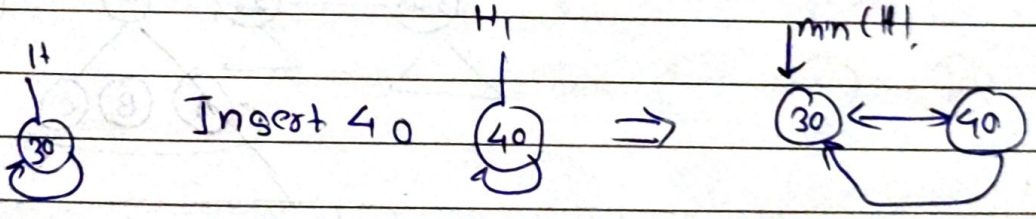
$\therefore P_i - P_{i-1} = 1.$

Amortized Cost = Actual Cost = $O(1)$

* Binomial Heap:

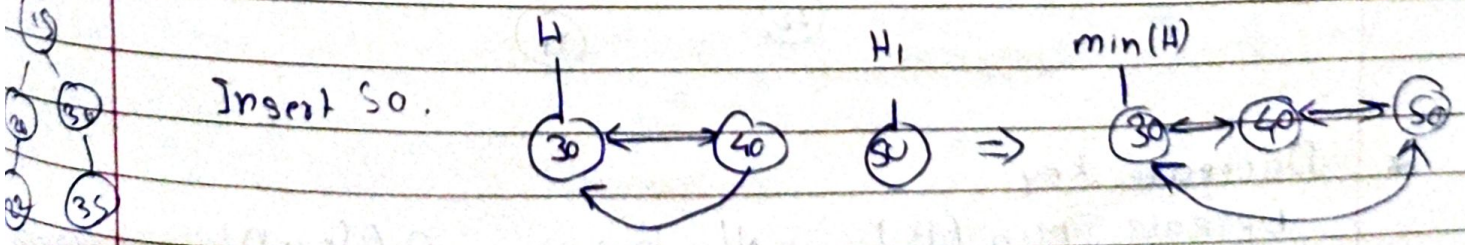


2^{degree} = No of Nodes (degree: Highest degree of any node)



During insert all elements will be added to root Cycles.

And it is trivial operation with complexity $O(1)$



5. Delete min

This operation is non trivial. It will cost order of log n.

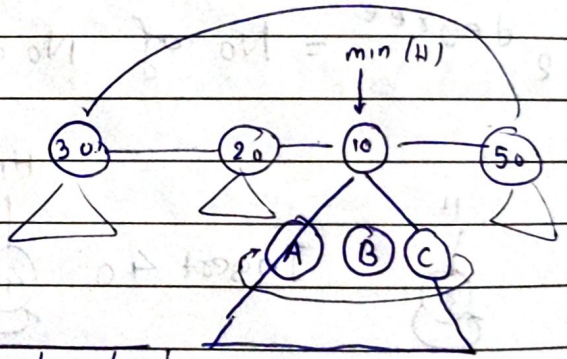
Fibonacci Heap.

It deletes min element in the heap and readjust head.

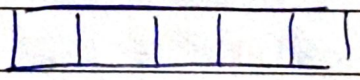
Steps:

1. Remove the Node with min key from the root cycle.
2. Merge the root cycle with the cycle of children of the Node.
3. while two roots r_1, r_2 have same degree
   do link r_1, r_2
   end while.
4. Adjust the minimum key

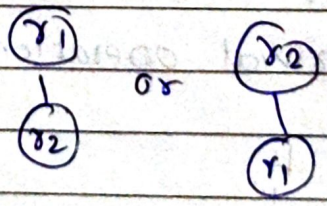
Decrease_key: (H, x, Δ)



Array A[ ] :



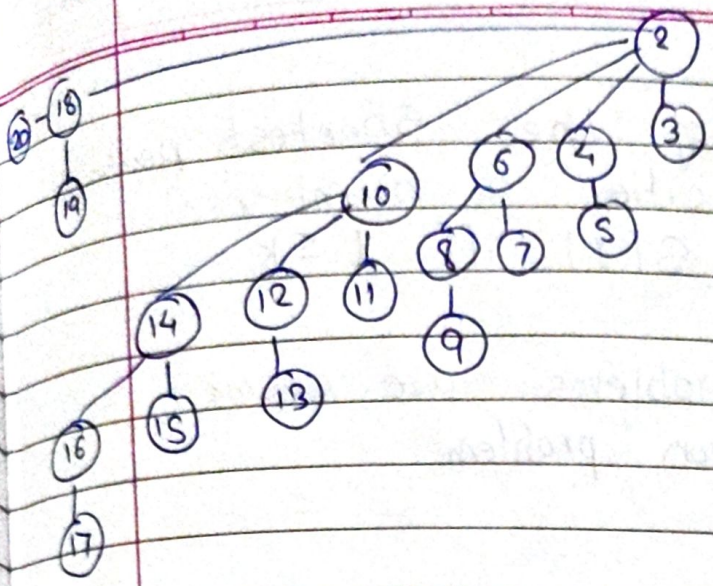
r_1, r_2 → Link.



Decrease_key

Delete Min (H) will have O(log n)

Decrease_key (H, x, Δ) : H is Some pointer to node we want to decrease x by Δ



* NP-Completeness:

When the every problem in the domain of computer Science can be Solved in polynomial time Complexity.

n^c vs c^n

Polynomial exponential.

There are certain problems example n! for which we dont know whether problem can be Solved in polynomial time or not.

If problem A can be Solved in polynomial time then problem B, C ... Can also Solved in polynomial time

This process is called reduction.

* Definition of Problem:

An abstract problem is a relation

$I \times S$

I = {set of problems

S = {set of Solutions using some particular tool}

An instance is a weighed graph and an pair of vertices.

Solution = Sequence of vertices from S to t or of number representing length of path.

* Decision Problems:

$I \times S$

$S = \{0, 1\}$

{No, Yes}

This relation is called function

The decision version of the shortest path problem also specifies a number A and Solution is S(i) iff $\lambda \leq k$.

Convert the decision problems are easier compared to optimization problem.

Optimization		Decision
Hard	$\Longrightarrow$	Easy
V Hard	$\Longleftarrow$	Hard

If optimzation pr

- Inside the computer system every problem and Solution is represented as binary string
- An encoding is a map

$I \longrightarrow \{0,1\}^*$ # String of 0's and 1's

instance Map.

A concrete decision problem is a decision problem with

$I = \{0,1\}^*$