§1 Lecture 03-12

§1.1 Amortized Time

Definition 1.1 (Amortized Time)**.** Amortized Time := Actual Time $+\Delta\phi_{\text{Potential}}$

$$\phi \geq 0$$

Note 1.2. If $\phi(\text{before}) = 0$, then Amortized time $\geq$ Actual time.

Example 1.3 (Lazy Delete)

- Fake / marked items.
- At 50% "fake", reconstruct the data structure.

Let $\phi = 2\cdot$ # Fake Elements. Then:

	Actual	\Delta \phi	AM
Insert	logn	0	log n
Lazy Delete	1	2	3
Reconstruct	n	-n	0

Actual time: <u>Cost</u> of t operations (Insert, Delete) starting from an empty tree

$$\leq t \log t + 3t$$

Example 1.4 (Counting to n in the bit model)

Let ϕ be the number of ones.

Actual time take $k+1$ where k is the number of ones before the first zero reading from left to right.
$\Delta\phi = -k + 1$.
So Amortized time $= 2$ for each increment.

Therefore total amortized time is $= 2n$, so actual time is $\leq 2n$.

§1.2 Fibonacci Heap

Operations.

	Binary Heap	Fibonacci Heap (Amortized)	AM
Insert	logn	1	log n
Delete	logn	$\log n$	3
Delete Min	logn	$\log n$	0
Decrease Key	logn	1	0
Meld (Join)	logn * logm	1	0

General Stucture is a bunch of double linked lists connected to another another, along with a "root list" and pointer to the min in the "root list".

1. Insert(x, F). Add another "subtree" to the root list.

2. Meld(F_1, F_2). Link the root lists of the two trees together.
3. Decrease key (x, k, F). Insert x tree into root list and change the key.

We will ensure that max degree $= O(\log n)$. Proof later. Let $\phi = \alpha$ size of root list $+ \beta$ number of marked items.