**PROBALISTIC DATA STRUCTURES:**

**Hashing:**

Suppose we have 2 sets A and B

We randomly pick a function from A to B

Look at all possible function from A to B

We want a data structure that can add an element, delete an element, and search for an element efficiently

To do this, store S in a hash table

When h(x) = h(y), we have a collision, which we want to avoid

Suppose we have a set S, |S| = n, and table T, |T| = m

Ideally, n ~ m

H is one-one on S if for all x != y that are elements of S, h(x) != h(y)