USA Tutor

Maps and Sets

By David Yang

What is a Set

An Array of Stuff (any type) that contains no duplicates

You cannot index a set

You can do .contains() in O(1) time, .add() in O(1) time,

.remove() in O(1) time

C++ set stuff

For .contains(), you have to do

```
    // if set contains the key
    If(set.find(key) !=set.end()) {}
```

For .add

```
    set.emplace(key)
```

For .remove()

```
    set.erase(key)
```

Java set stuff

For .contains(), just do set.contains(key)

For .add(), just do .add(key)

For .remove, just do .remove(key)

What kinds of sets are there?

unordered_set<int>/HashSet<Integer>

a set that hashes the input that you give it.

This runs in O(1) time for every operation (also worst case O(N))

set<int>/TreeSet<Integer>

a set that sorts things with a balanced binary tree

This runs in O(logN) time for every operation

Why use a set?

If you wanted to see if a string/pair/any object existed

In really fast time

Add it in really fast time

And remove in really fast time…

Buy a set! Only 1Bessiecoin for a set (just use the set)

What is a Map?

A hotel room.

Has a key, and a value for that key.

All keys are unique, all values do not have to be.

The keys are hashed so you can look stuff up in O(1) time

Every operation, .emplace(a,b), .erase(a), .contains(a) runs in O(1)

C++ Map Stuff

To check for key exists

```
if(map.find(key)!=map.end())
```

To insert a key:value pair

```
map.emplace(key,value)
```

To remove a key:value pair

```
map.erase(key)
```

Java Map Stuff

map.containsKey(key)

map.put(key,value)

map.remove(key)

Where Am I (USACO 2019 Dec Bronze)

We are given N<=100 length string

Find the minimum substring length K such that no 2 substrings with length K are equal

We can do this with brute force checking, or…

Sets!

Why use a set?

Instead of having to iterate over all of our keys one by one

We can just call a function, and the function runs really fast!

Initializing the String

```
set<string> s;
for(int k=1; k<K; k++){
        string temp="";
        for(int i=0; i<k; i++){
                temp+=str[i];
        }
        s.emplace(temp);
}
```

Checking for validity

```
for(int k=1; k<K; k++){
        … s.emplace(temp);
        bool valid=true;
        //take every element except for first
        for(int i=k; i<N; i++){
                temp= temp.substr(1, k-1);
                temp+=str[i];
                //temp existed before, illegal
                if(s.find(temp)!=s.end()){
                        valid=false; break;
                }
        }
}
```