Sets

Meant to model mathematical sets:

in particular, $S \subseteq U$ where

U = all values of a particular datatype.

E.g. $U = \{0, 1, 2, 3, 4, 5, 6, 7\}$.

$S = \{1, 5\}$

$S \subseteq U \equiv \forall x \in S,\ x \in U$

("for all")

"S is a subset of U"

"x is a member of S".

Basic operations: union ($\cup$) & intersection ($\cap$)

How to represent a set, say in a program?

Say the universe set U is all 16-bit integers:

$U = \{0, 1, \ldots, 2^{16}-1\}$.

How to represent any $S \subseteq U$?

Want to quickly be able to answer questions of the form "is $x \in S$?"

What about a vector<bool> S with size $= 2^{16}$?

$x \in S \iff S[x] == \text{true}$.

So if $S = \{1, 4\}$, (math world)

vector S = [false, true, false, false, true, false, false...]
0 1 2 3 4 5 ...

(C++ world)

Answering "$x \in S$?" very fast!

Downside: could require lots of memory:

size = |U|.

Even for datatype int, this is probably a bad idea.

what if U is large?

Could use vector<int> and store the values: if $S = \{1, 4\}$. (math)

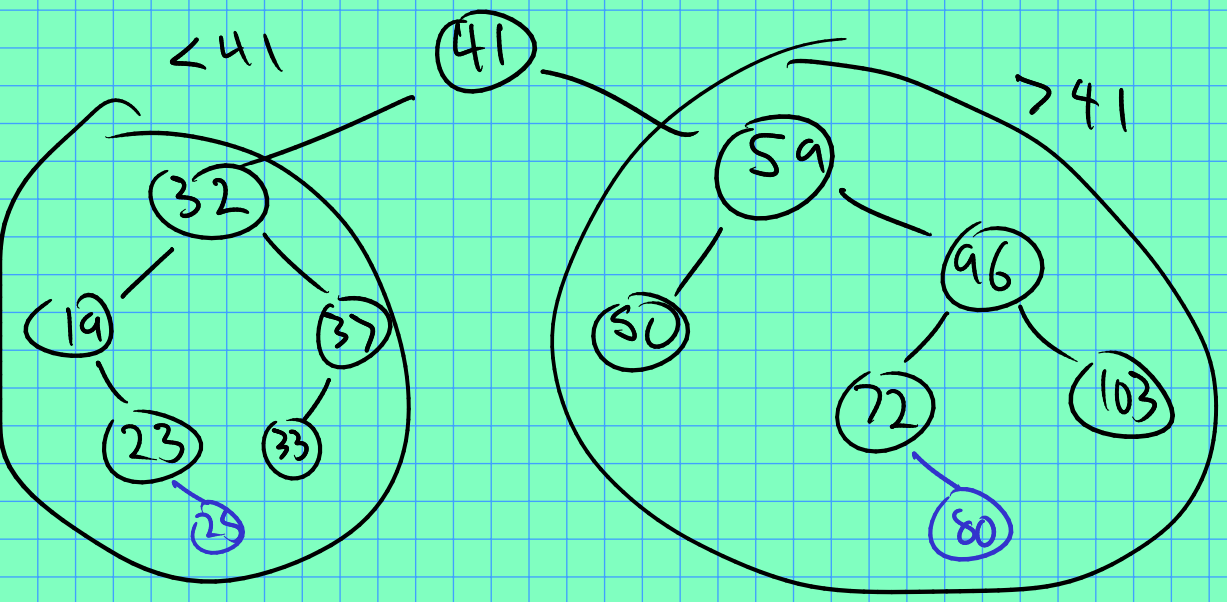
$S = [1, 4]$ (C++ vector).

Issues: might be difficult to answer "$x \in S$?" questions. Have to look through entire vector!

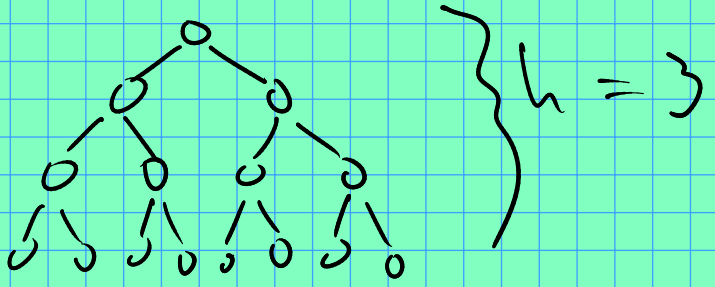
Could sort the vector, but this is difficult to maintain if elements are added / removed.

STL <set> is a nice solution:

elements are stored in a tree structure:



≈ cost for search (if $|S| = n$)?



total # values $= 2^{h+1} - 1$

cost $\approx h$, but $n \approx 2^h$

So $h \approx \log_2 n$.

C++ details: comparison w/ vector

vector:	set:
v.push_back(x)	S.insert(x)
v.pop_back()	S.erase(x)
v[i]	no (efficient) equivalent

```
for(i=0; i<V.size(); i++)
    cout << V[i];          //vector
```

```
for(i=s.begin(); i!=s.end(); i++)
    cout << (*i);
```

(type for i = set<int>::iterator)