Note Title 22-10-2018

CSE100

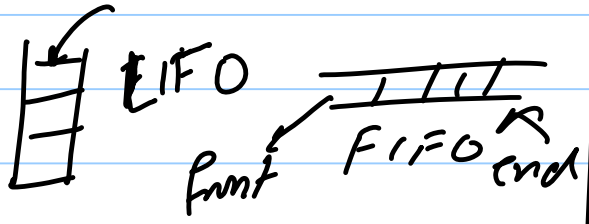
Oct 22, 2018

Last Couple of Classes:-

(1) Stacks / queues

(2) Debugger (gdb —>)

ADT



Make-up exam:-

Minor 1 / Minor 2

Friday Oct 26

5:30 pm - 6:30 pm

Venue:- TBA

Syllabus:-

Everything

(covered up to

Minor 2

(from beginning)

	Stack<type> s; (Stanford)	stack<type> s; (C++) # include <stack>
①	s.isEmpty()	s.empty()
②	s.peek()	s.top()
③	s.push(value)	s.push(value)
④	s.pop()	s.pop() → does not return a value.
⑤	s.size	s.size()

Queue:		# include <queue>
	q.isEmpty()	q.empty()
	q.peek()	q.front()
	q.dequeue()	q.pop() → does not return a value
	q.enqueue(value)	q.push(value)
	q.size()	q.size()

C++ STL

Sets: - ADT

① No order among elements

② No duplicates

{2, 2, 3, 1, 1, 12}

{1, 2, 3}

→ Where would they be useful?

→ Website unique users

→ Dictionary

C++: `set<type> s;`

Plan for Rest of the semester:

(1) Sets / Maps

(2) Recursion / Backtracking

(3) Some additional topics

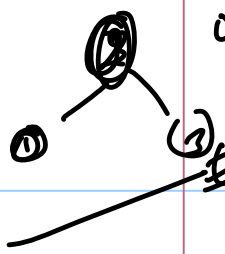
- find(value)
- insert / add element (value)
- remove / erase (value)
- size()

Example -

```
set<string> friends;
friends.insert("abc");
friends.insert("xyz");
for(string myfriend : friends) {
    cout << "Hi" << myfriend << endl;
}
```

Goes over the elements in alphabetic order

Can't modify or change the set

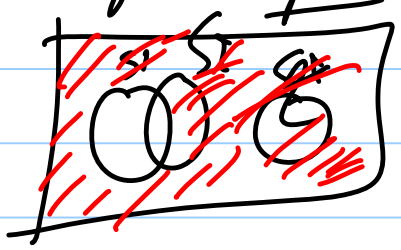


ole to ignore

S.insert("abc");
S.insert("xyz");

S2.insert("xyz");
S2.insert("cbl");

Operation of sets:-



→ Union
→ Intersection
→ Diff

{"xyz", "abc"} → "abc"

	Standard	C++ (STL)	Time Complexity
①	s.add(value)	s.insert(value)	O(log N)
②	s.clear()	s.clear()	O(N)
③	s.Contains(value)	s.find(value) != s.end()	O(log N)
④	s.equals(s2)	s == s2	O(N)
⑤	s.first()	*s.begin()	O(log N)

Stanford	C++	Time
s.isEmpty()	s.empty()	O(1)
s.isSubsetOf(s1)	No equivalent fn	~~O(N)~~ ~~O(N)~~ O(max(N,M))
~~s.~~ ~~s.~~ s.remove(value)	s.erase(value)	O(log N)
s.size()	s.size()	O(1)
s.toString()	No equivalent fn	O(N)

{1, 2, 3, 10, 15}
{"abc", "xyz"}

Thinking about what to use when?

ADT:- vectors, sets, stack, queue, grids

→ Question:- 2-D data? 1-D Data
↳ Grid

→ Question:- Care about duplicates or not?

Care about duplicates:- Don't Care → sets

↳ Vectors (Access to any Direct element)

→ Stack/Queue

52

"2017CS12056"	10	7	20	5

A	A-	B	B-	C	C-
80	72	64	54	45	

Maps ADT

↳ ~~Set~~ set of <key, value> pairs

unique ← key, → value

prevent multiple times

→ Maps:- Access the elements using key

"2017CS1036" ~~B~~

map<type, type> m;

ordered by key (value)

← map<string, double> m;

```
map.put(key, value);
map[key] = value;
```

② `m.get(key)` → exception if key is not there

`m[key]` → if key is not there, insert a default value for key (set & return it)

③ `m.remove(key)` → Removes key & corresponding value from map (nothing happens if key is not there)

Map<type, type> m;

map<type, type> m;

Member Functions:-

	Stanford	C++ STL
①	m.put(key, value)	m[key] = value;
~~②~~	m[key] = value	m[key] = value;
②	m.get(key)	~~m[key]~~ m.at[key]
	returns the value for key exception if not present	DIFF
	m[key]	m[key] — if key is not present, creates the key, value pair with default value

(3)

m.remove(key)	m.erase(key)
	↳ nothing happens if key is not there
(4)	
↳ m.lower_bound(key)	m.find(key) != m.end()
	→ True if key is present else false