CS 225

Data Structures

February 1 – Array List Take 2

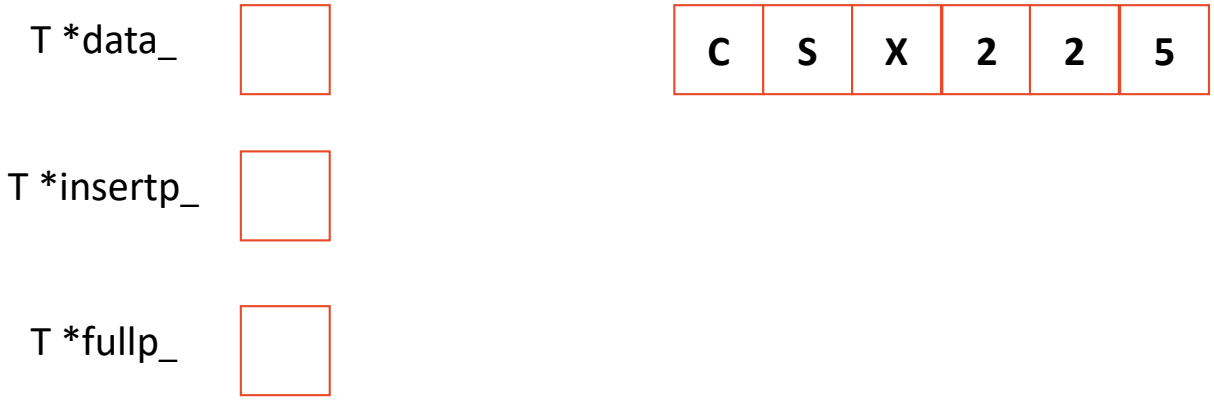
G Carl Evans

List.h

```
 1 #pragma once
 2
 3 template <typename T>
 4 class List {
 5 public:
 …
18   void pushback(const T &data);
 …
25 private:
26   T *data_;
27   T *insertp_;
28   T *fullp_;
29
30   void _addspace();
 …     /* --- */
   };
```

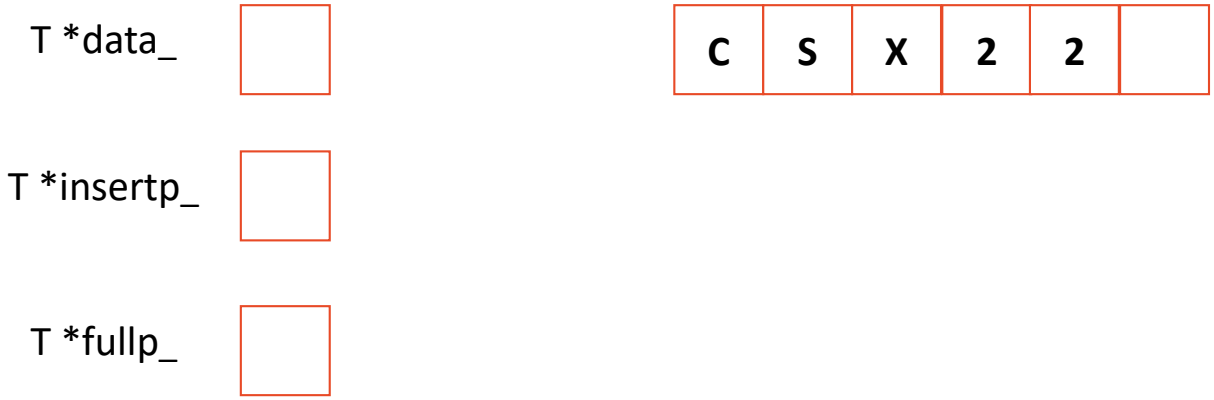
Array Implementation

_addspace():



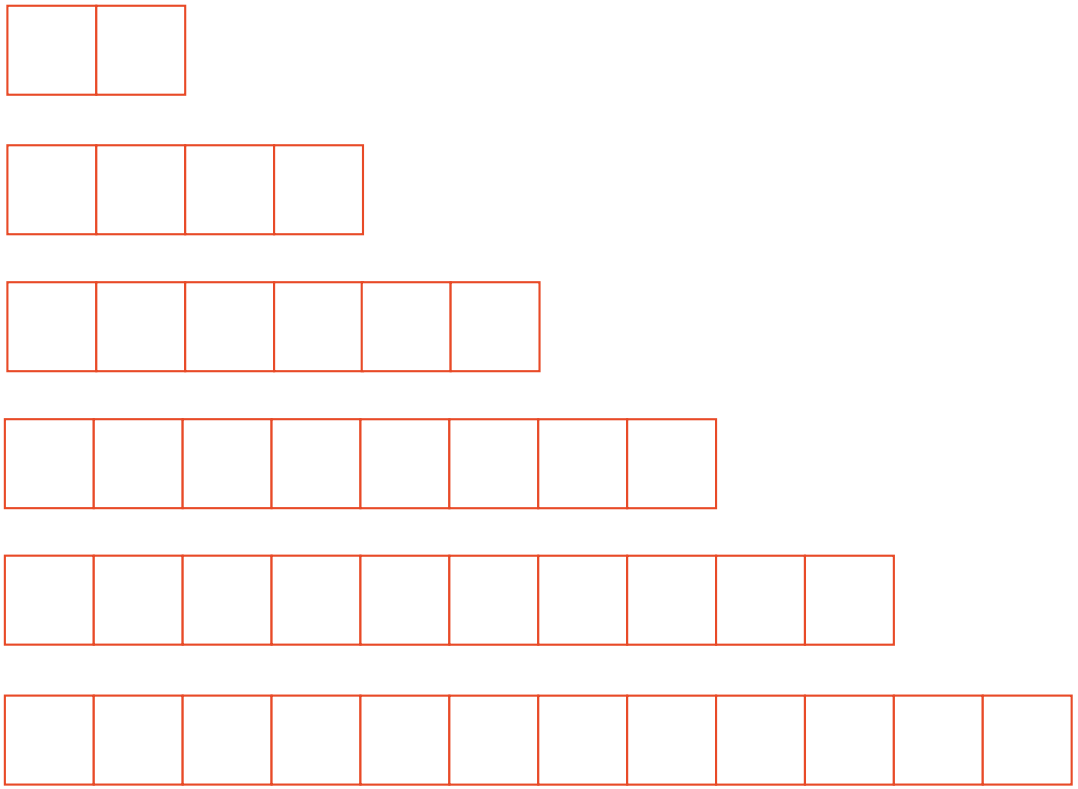
Array Implementation

pushback(data):



Amortized Analysis

Resize Strategy: +2 elements every time



Resize Strategy: +2 elements every time

Queue ADT

- [Order]:
- [Implementation]:
- [Runtime]:

Stack ADT

- [Order]:
- [Implementation]:
- [Runtime]:

Queue.h

```
 1  #pragma once
 2
 3  template <typename T>
 4  class Queue {
 5    public:
 6      void enqueue(T e);
 7      T dequeue();
 8      bool isEmpty();
 9
10    private:
11      T *items_;
12      size_t capacity_;
13      size_t size_;
14  };
15
16
17
18
19
20
21
22
```

What type of implementation is this Queue?

How is the data stored on this Queue?

Queue.h

```
1  #pragma once
2
3  template <typename T>
4  class Queue {
5    public:
6      void enqueue(T e);
7      T dequeue();
8      bool isEmpty();
9
10   private:
11     T *items_;
12     size_t capacity_;
13     size_t size_;
14 };
15
16
17
18
19
20
21
22
```

What type of implementation is this Queue?

How is the data stored on this Queue?

```
Queue<int> q;
q.enqueue(3);
q.enqueue(8);
q.enqueue(4);
q.dequeue();
q.enqueue(7);
q.dequeue();
q.dequeue();
q.enqueue(2);
q.enqueue(1);
q.enqueue(3);
q.enqueue(5);
q.dequeue();
q.enqueue(9);
```

Queue.h

```
1  #pragma once
2
3  template <typename T>
4  class Queue {
5    public:
6      void enqueue(T e);
7      T dequeue();
8      bool isEmpty();
9
10   private:
11     T *items_;
12     size_t capacity_;
13     size_t size_;
14 };
15
16
17
18
19
20
21
22
```


```
Queue<char> q;
...
q.enqueue(m);
q.enqueue(o);
q.enqueue(n);
...
q.enqueue(d);
q.enqueue(a);
q.enqueue(y);
q.enqueue(i);
q.enqueue(s);
q.dequeue();
q.enqueue(h);
q.enqueue(a);
```