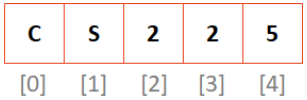
CS 225

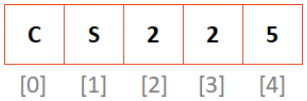
#11: Stacks and Queues

February 9, 2018 · *Wade Fagen-Ulmschneider*

Array-backed List - Implementation Details:

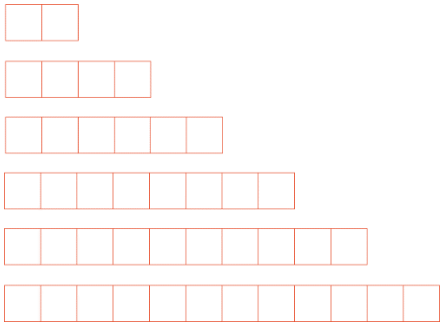


1. What is the running time of `insertFront()`?

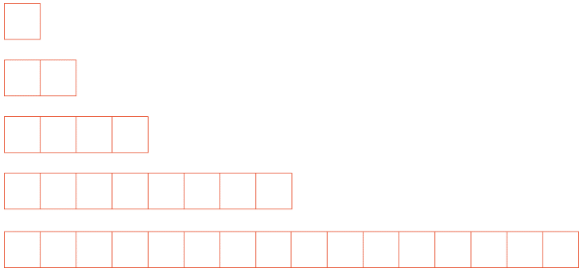


2. What is our resize strategy?

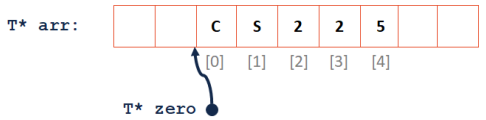
Resize Strategy #1:



Resize Strategy #2:



3. What is the running time of get()?



	Singly Linked List	Array
Insert/Remove at **front**		
Insert after a **given** element		
Remove after a **given** element		
Insert at **arbitrary** location		
Remove at **arbitrary** location		

Stack ADT

Function Name	Purpose

Queue ADT

Function Name	Purpose

Stack and Queue Implementations

Stack.h

```
1  #ifndef STACK_H_
2  #define STACK_H_
3
4  #include "List.h"
5
6  template <typename T>
7  class Stack {
8    public:
9      void push(T & t);
10     T & pop();
11     bool isEmpty();
12
13   private:
14     List<T> list;
15
16 };
17
18 #endif
```

Stack.cpp

```
1  #include "Stack.h"
2
3  template <typename T>
4  void Stack::push(T & t) {
5    list_.add(t, 0);
6  }
7
8  template <typename T>
9  T & Stack::pop() {
10   return list_.remove(0);
11 }
12
13 bool Stack::isEmpty() {
14   return list_.isEmpty();
15 }
```

Three designs for data storage in data structures:

1. T & data

2. T * data

3. T data

Implication of Design

1. Who manages the lifecycle of the data?

2. Is it possible to store a NULL as the data?

3. If the data is manipulated by user code while stored in our data structure, are the changes reflected within our data structure?

4. Speed

	Storage by Reference	Storage by Pointer	Storage by Value
Lifecycle management of data?			
Possible to insert NULL?			
External data manipulation?			
Speed			

CS 225 – Things To Be Doing:

1. Programming Exam A starts Feb. 13 *(next Tuesday)*
2. MP2 due Feb. 12 *(next Monday)*
3. lab_inheritance due Sunday
4. Daily POTDs