CS 225

Data Structures

February 15 – List Implementation

G Carl Evans

List ADT to C++ Interface

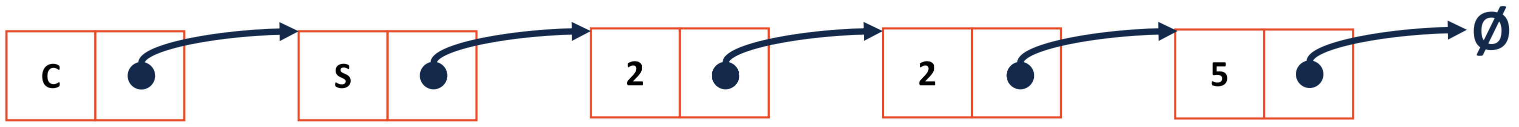
- **Insert – `void insert(const T &data);`**
- **Delete – `void delete();`**
- **Get Data – `T getData() const;`**
- **Is Empty – `bool isEmpty() const;`**
- **Create Empty List – `List();`**

List Implementations

1.

2.

Linked Memory



List.h

```
class ListNode {
  T  data;
  ListNode * next;
  ListNode(T & data) : data(data), next(NULL) { }
};
```

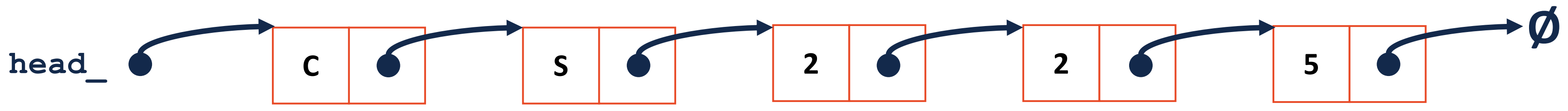
List.h

```
#pragma once

template <typename T>
class List {
  public:
    /* ... */

  private:
    class ListNode {
      public:
        T data;
        ListNode * next;
        ListNode(T & data) :
        data(data), next(NULL) { }
    };

    ListNode *head_;


};
```

List.hpp

```
#include "List.h"

template <typename T>
void List::insertAtFront(const T& d)  {







}
```

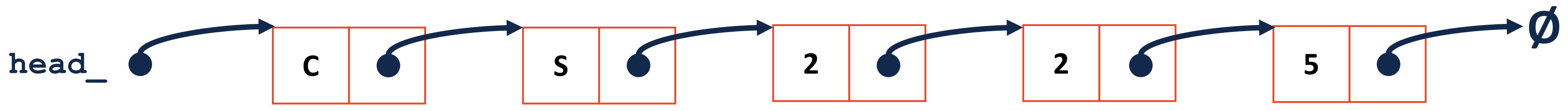
Linked Memory



List.hpp

```
template <typename T>
typename List<T>::ListNode *&
  List<T>::_index(unsigned index) {


}
```

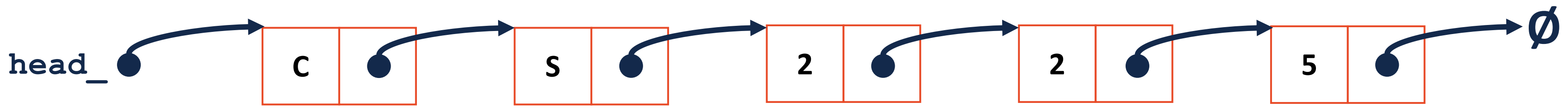
Linked Memory



List.hpp

```
// Iterative Solution:
template <typename T>
typename List<T>::ListNode *& List<T>::_index(unsigned index) {
  if (index == 0) { return head; }
  else {
    ListNode *thru = head;
    for (unsigned i = 0; i < index - 1; i++) {
       thru = thru->next;
    }
    return thru->next;
  }
}
```

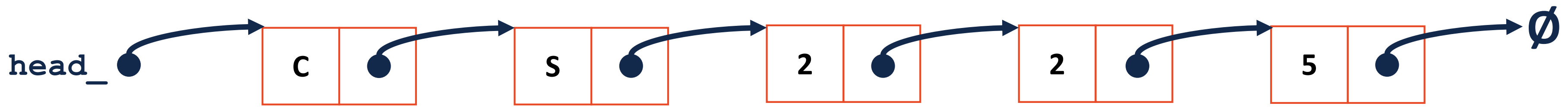
Linked Memory



List.cpp

```cpp
template <typename T>
T & List<T>::operator[](unsigned index) {









}
```

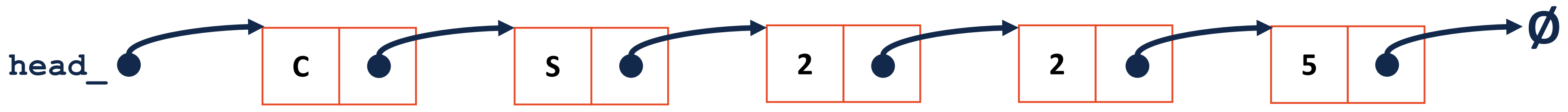
Linked Memory



List.cpp

```
template <typename T>
void List<T>::insert(const T & t, unsigned index) {







}
```

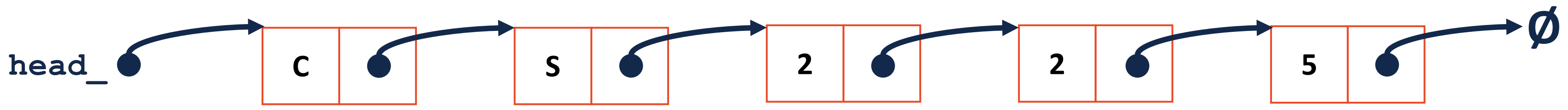
Linked Memory



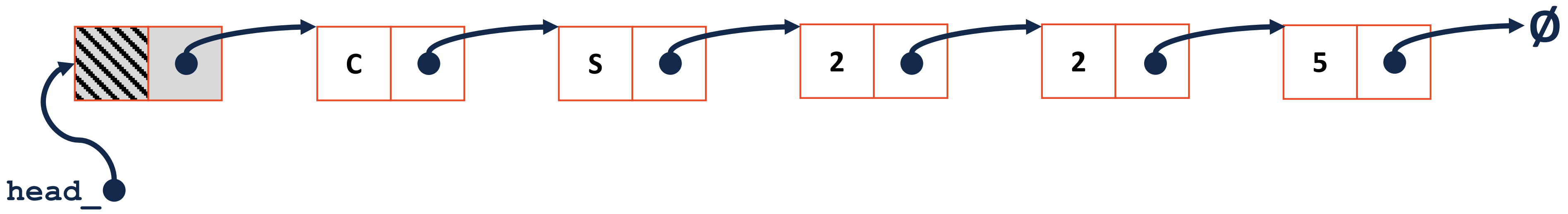
List.cpp

```
template <typename T>
T List<T>::remove(unsigned index) {








}
```

Linked Memory



Sentinel Node



List Implementations

1. **Linked List**

2.

List.h

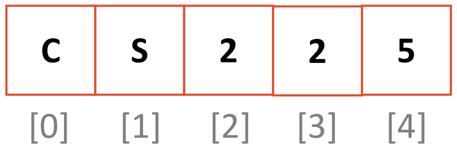
```
#pragma once

template <typename T>
class List {
  public:
     /* ... */
  private:
















};
```

Array Implementation

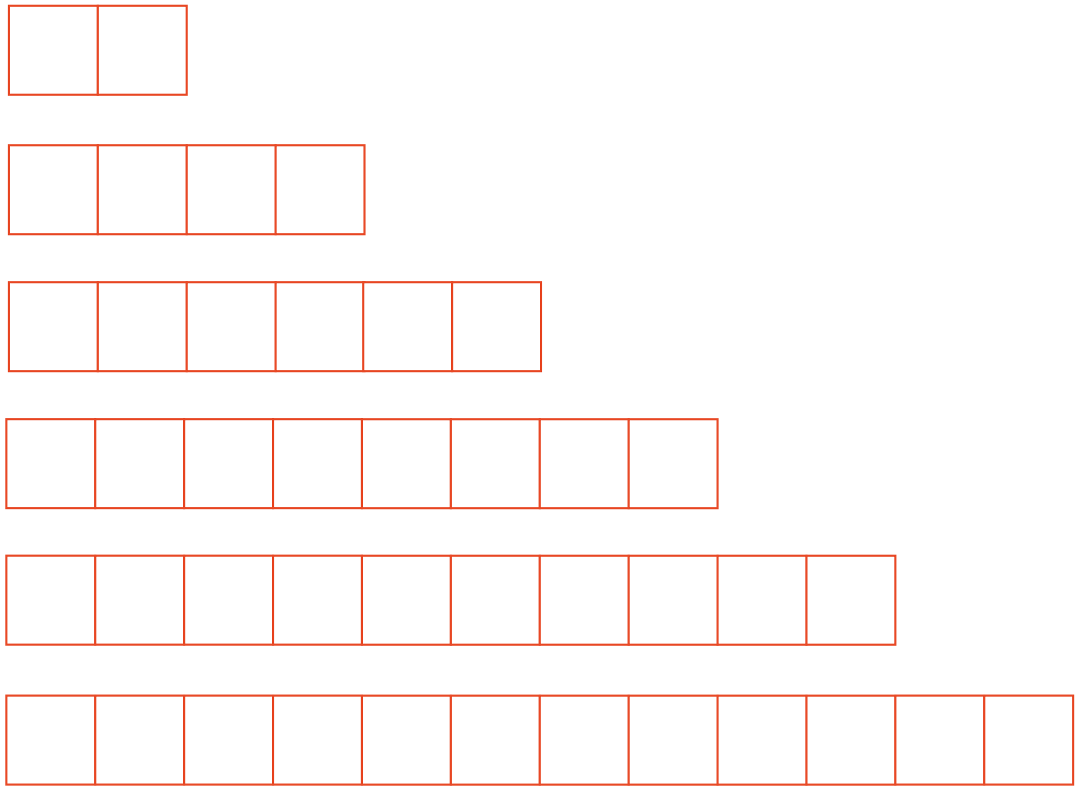
C	**S**	**2**	**2**	**5**
[0]	[1]	[2]	[3]	[4]

Array Implementation

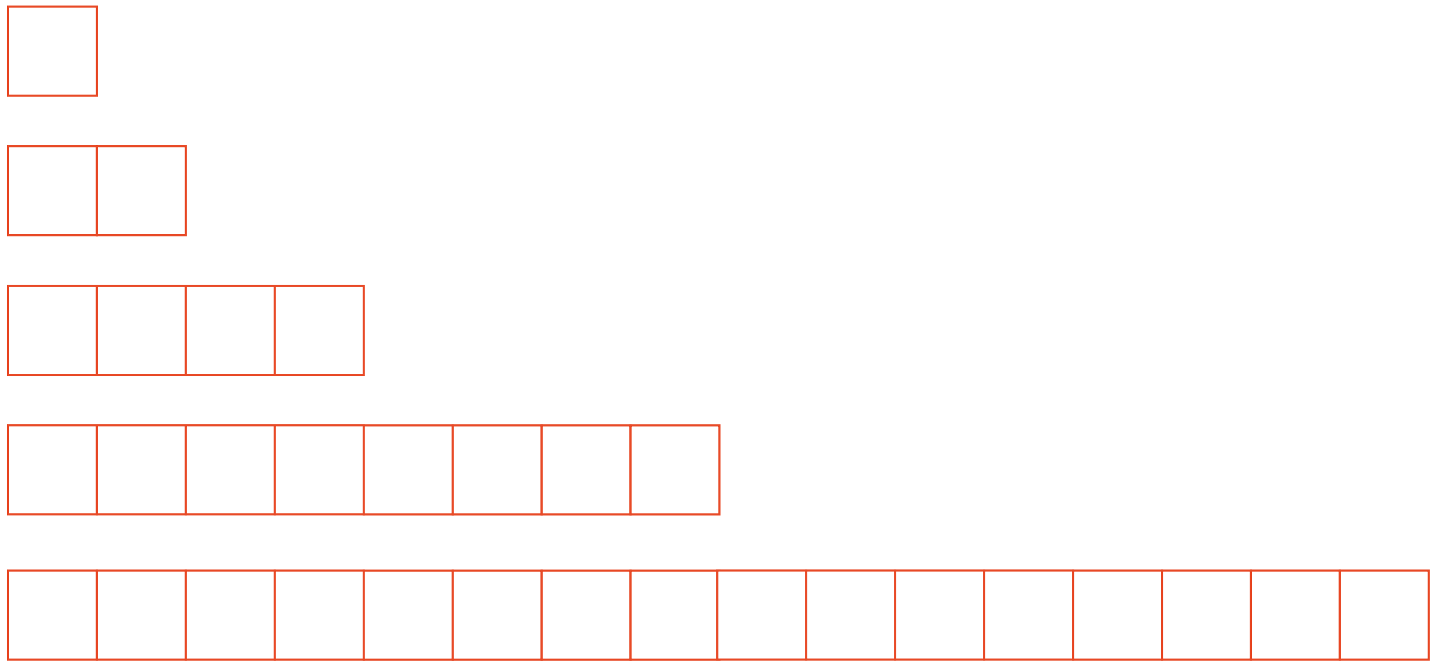
insertAtFront:



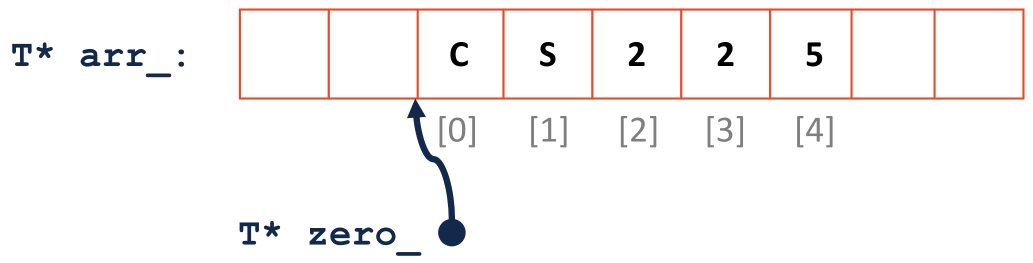
Resize Strategy – Details



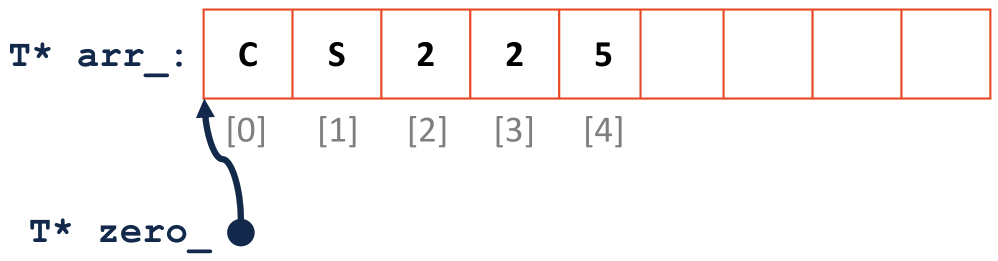
Resize Strategy – Details



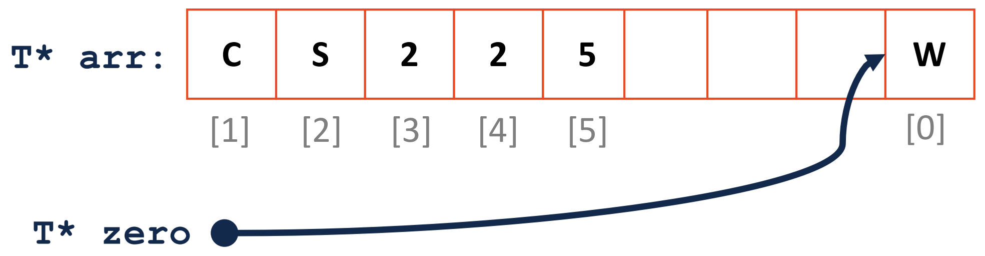
Array Implementation



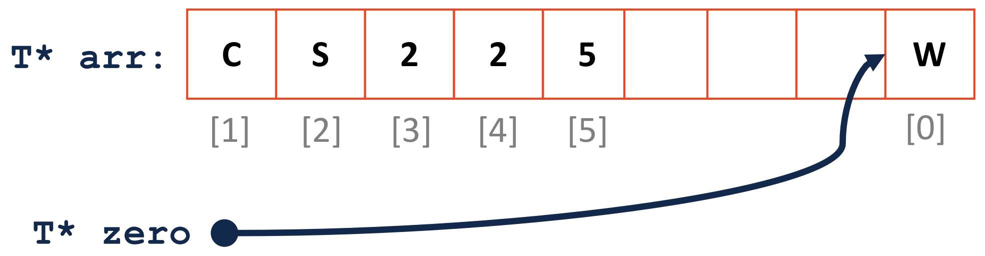
Array Implementation



Array Implementation



Array Implementation



Array Implementation

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