# Stacks and Queues

Chapter 18

## Stacks

- A stack is a data structure that stores and retries items in a first-in-last-out (FILO) or last-in-first-out (LIFO) manner

## Stacks

- A stack is a data structure that stores and retries items in a first-in-last-out (FILO) or last-in-first-out (LIFO) manner

- This would be the opposite of what you would expect from a line at a store

## Stacks

- There are two main types of stacks

## Stacks

- There are two main types of stacks
    - Static Stack
        - Consists of a static size
        - Often implemented using an array

## Stacks

- There are two main types of stacks
  - Static Stack
    - Consists of a static size
    - Often implemented using an array
  - Dynamic Stack
    - There is no max size
    - Often implemented using a Linked List

## Stacks Operations

- Since a Stack is mainly concerned with the top item, its operations are pretty simple

## Stacks Operations

- Since a Stack is mainly concerned with the top item, its operations are pretty simple
  - push
    - Add an item to the stack

## Stacks Operations

- Since a Stack is mainly concerned with the top item, its operations are pretty simple
    - push
        - Add an item to the stack
    - pop
        - remove an item from the stack

## Stacks Operations

- There are other useful operations for a Stack
  - top
    - Returns the item on the top of the stack
  - capacity
    - For static stacks, returns the maximum number of items allowed

## Stacks Operations

- There are other useful operations for a Stack
    - top
        - Returns the item on the top of the stack
    - capacity
        - For static stacks, returns the maximum number of items allowed
    - isEmpty
        - Returns true if the stack is empty

## Stacks Operations

- Unlike dynamic stacks, a static stack push can fail

## Stacks Operations

- Unlike dynamic stacks, a static stack push can fail
    - A good way to handle this is through an exception

## Template Stacks

- Similar to our other containers and structures, both the dynamic and static stacks can made into a template stack

## Stacks in the STL

- In the STL, we have seen vectors and lists already

- The STL also contains a stack

## Stacks in the STL

- The STL stack can be implemented as a vector or list

## Stacks in the STL

- The STL stack can be implemented as a vector or list
  - The stack is said to adapt to another class

## Stacks in the STL

- The STL stack can be implemented as a vector or list
    - The stack is said to adapt to another class

```
stack< int, vector<int> > intStack; // vector stack
stack< int, list<int> > intStack; // list stack
stack< int> intStack; // Deque stack
```

## Queue

- A queue is a data structure that stores and retrieves items in a first-in-first-out (FIFO) manner

## Queue

- A queue is a data structure that stores and retrieves items in a first-in-first-out (FIFO) manner

- This is what you might expect from a line at a store

## Queue

- Similar to Stacks, a Queue consists of two types

## Queue

- Similar to Stacks, a Queue consists of two types
    - Static Queue
        - Static size
        - Can be implemented with an array

## Queue

- Similar to Stacks, a Queue consists of two types
    - Static Queue
        - Static size
        - Can be implemented with an array
    - Dynamic Queue
        - No max size
        - Can be implemented with a Linked List

## Queue Operations

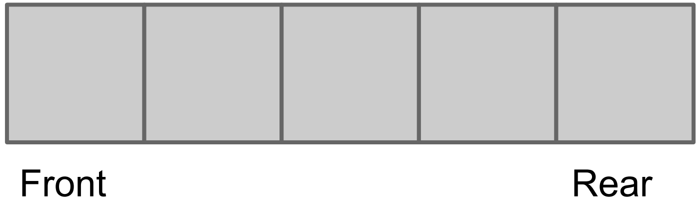
- Queue operations are simple
    - Enqueue
        - Add an item to the Queue

## Queue Operations

- Queue operations are simple
    - Enqueue
        - Add an item to the Queue
    - Dequeue
        - Remove an item from the Queue

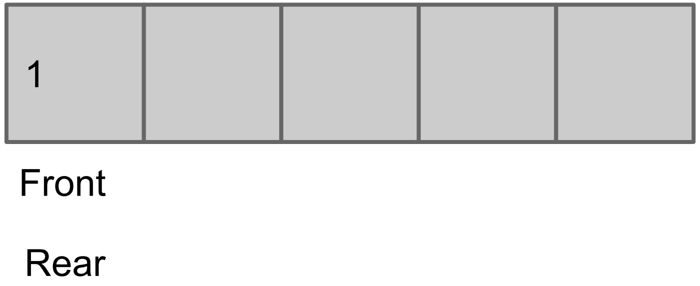
## Queue

- Example of a Static Queue



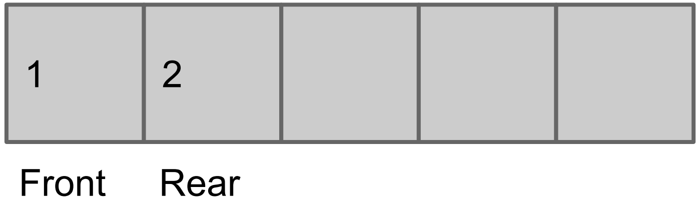
# Queue

- Example of a Static Queue
  - As items are added, they go from the front down



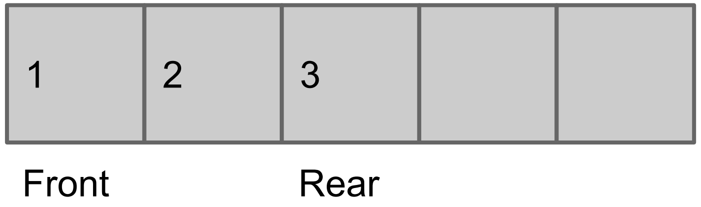
## Queue

- Example of a Static Queue
  - As items are added, they go from the front down



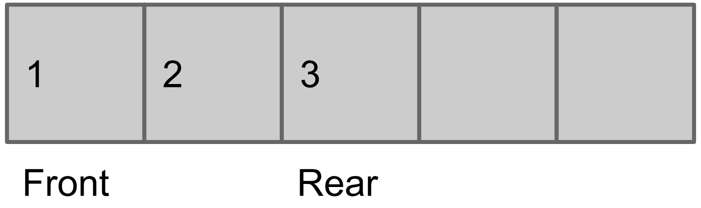
## Queue

- Example of a Static Queue
  - As items are added, they go from the front down



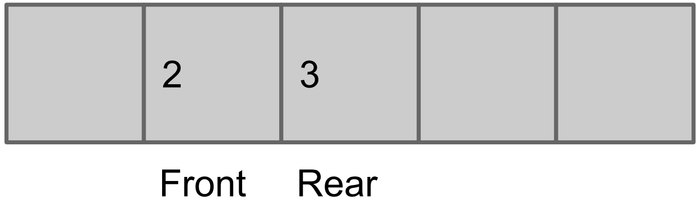
## Queue

- Example of a Static Queue
  - Items are removed from the front



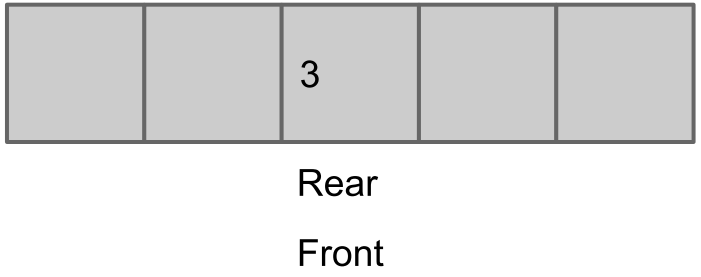
## Queue

- Example of a Static Queue
  - Items are removed from the front



## Queue

- Example of a Static Queue
  - Items are removed from the front



## Queue

- Because of how static Queues enqueue and dequeue, the have a specific problem

## Queue

- Because of how static Queues enqueue and dequeue, the have a specific problem
    - The array must be wrapped or the Queue can be used only until the front reaches the back of the array

## Queue

- Because of how static Queues enqueue and dequeue, the have a specific problem
  - The array must be wrapped or the Queue can be used only until the front reaches the back of the array
- This problem is solved with a circular array

## Queue

- Because of how static Queues enqueue and dequeue, the have a specific problem
  - The array must be wrapped or the Queue can be used only until the front reaches the back of the array
- This problem is solved with a circular array
  - rear = (rear + 1) % queueSize;

## Queue

- Since Static Queues do not grow, they may be overflowed or underflowed
  - Enqueue with no room
  - Dequeue with no item

## Queue

- Since Static Queues do not grow, they may be overflowed or underflowed
    - Enqueue with no room
    - Dequeue with no item
- This can be solved with exceptions

## Queue

- How about Dynamic Queues?

## Queues in the STL

- There are two Queue like structures in the STL
    - Queue
    - Deque (pronounced deck or deek)

## Queues in the STL

- The ‘queue’ container acts just like you would expect a queue to act

## Queues in the STL

- The ‘queue’ container acts just like you would expect a queue to act
  - Operations
    - push
      - push an item to the back of the queue

## Queues in the STL

- The ‘queue’ container acts just like you would expect a queue to act
  - Operations
    - push
      - push an item to the back of the queue
    - pop
      - remove the first element on the queue

## Queues in the STL

- The ‘deque’ container acts a bit different
    - This container is like a double sided queue

## Queues in the STL

- The ‘deque’ container acts a bit different
  - This container is like a double sided queue
  - It allows quick access to the front and back of the queue

## Queues in the STL

- The ‘deque’ operations are a bit different
  - push_back
    - places an item at the end of the queue

## Queues in the STL

- The ‘deque’ operations are a bit different
  - push_back
    - places an item at the end of the queue
  - pop_front
    - removes the first item in the queue

## Queues in the STL

- The ‘deque’ operations are a bit different
  - push_back
    - places an item at the end of the queue
  - pop_front
    - removes the first item in the queue
  - front
    - returns the first item on the queue