CSCI 132:
Basic Data Structures and Algorithms

Queues (Linked List implementation)

Reese Pearsall & Iliana Castillon

Fall 2024

toString() : Java method used to provide a string representation of an object

By default, it returns a string that consists of:

ClassName@hascode (memory address)

toString() : Java method used to provide a string representation of an object

By default, it returns a string that consists of:

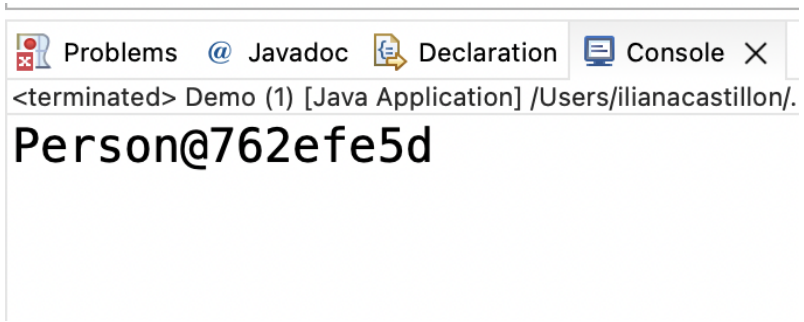
ClassName@hascode (memory address)

Person.java

```
public class Person {
  private String name;
  private int age;


  public Person(String name, int age) {
    this.name = name;
    this.age = age;
  }
}
```

Demo.java

```
public class Demo {

  public static void main(String[] args) {
    Person iliana = new Person("Iliana", 80);
    System.out.println(iliana.toString());
  }
}
```



toString() : Java method used to provide a string representation of an object

By default, it returns a string that consists of:

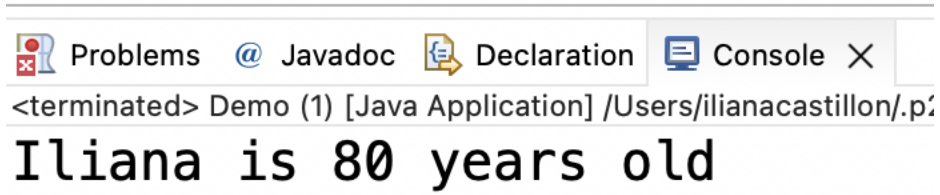
ClassName@hascode (memory address)

Person.java

```
public class Person {
  private String name;
  private int age;


  public Person(String name, int age) {
    this.name = name;
    this.age = age;
  }

  @Override
  public String toString() {
    return name + " is " + age + " years old";
  }
}
```

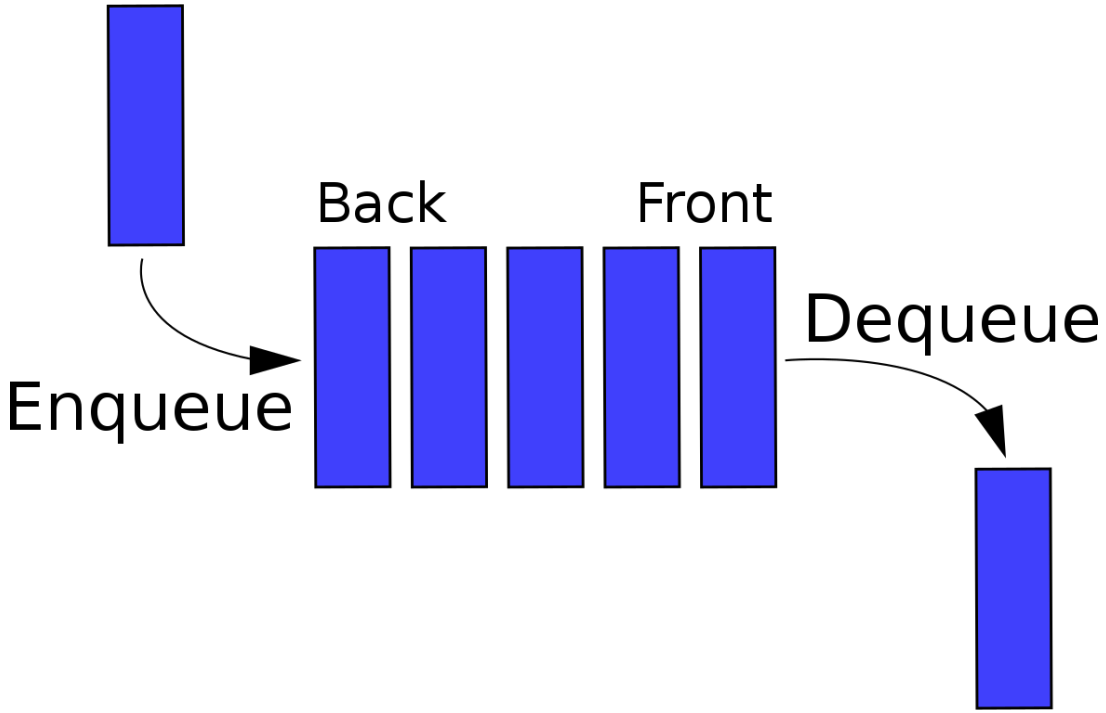
Demo.java

```
public class Demo {

  public static void main(String[] args) {
    Person iliana = new Person("Iliana", 80);
    System.out.println(iliana.toString());
  }
}
```

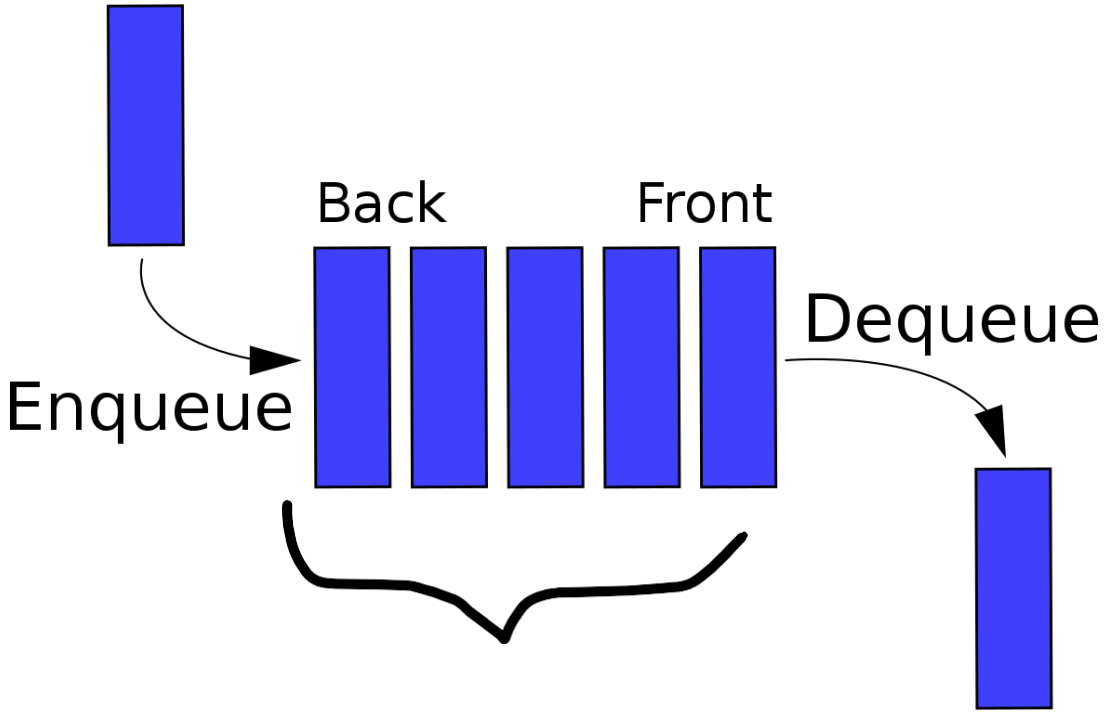


Announcements

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



Once again, we need a data structure to hold the data of the queue

- Linked List (today)
- Array (next week)

Elements get added to the **`Back`** of the Queue.

Elements get removed from the **`Front`** of the queue

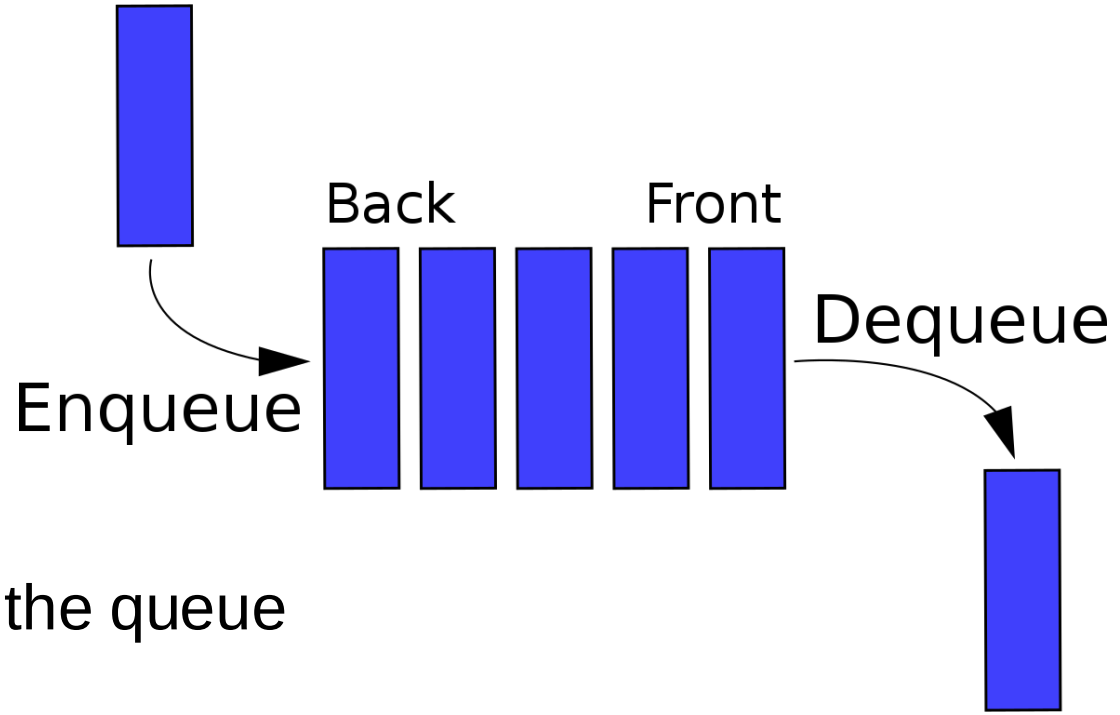
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion

The Queue ADT has the following methods:

Enqueue- Add new element to the queue

Dequeue- Remove element from the queue

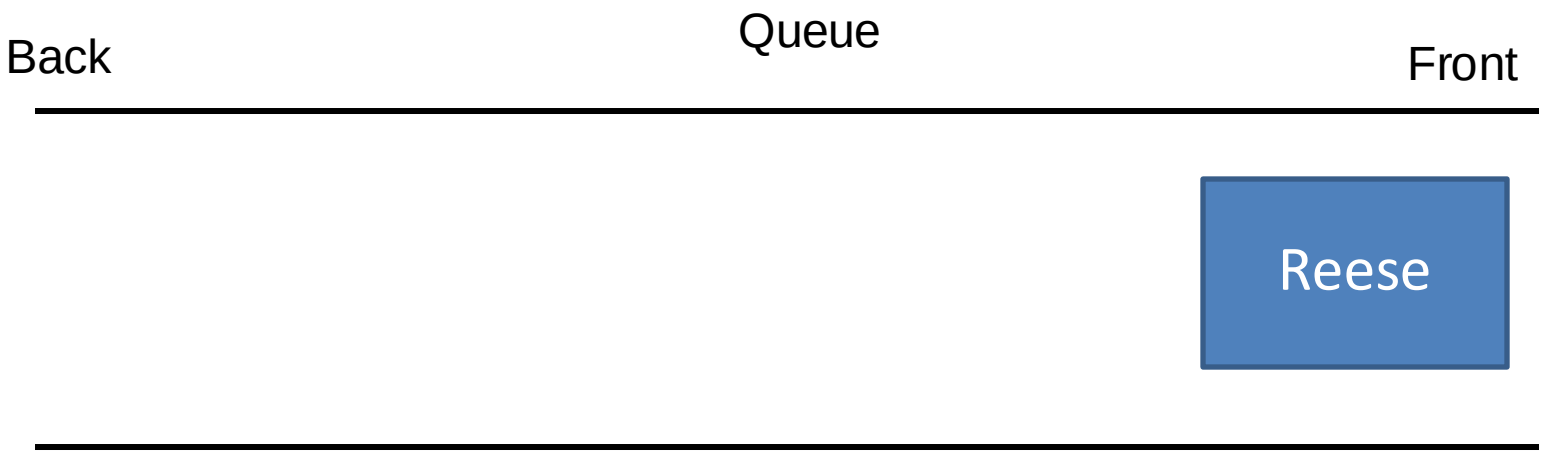
** Always remove the front-most element



Peek()- Return the element that is at the front of the queue

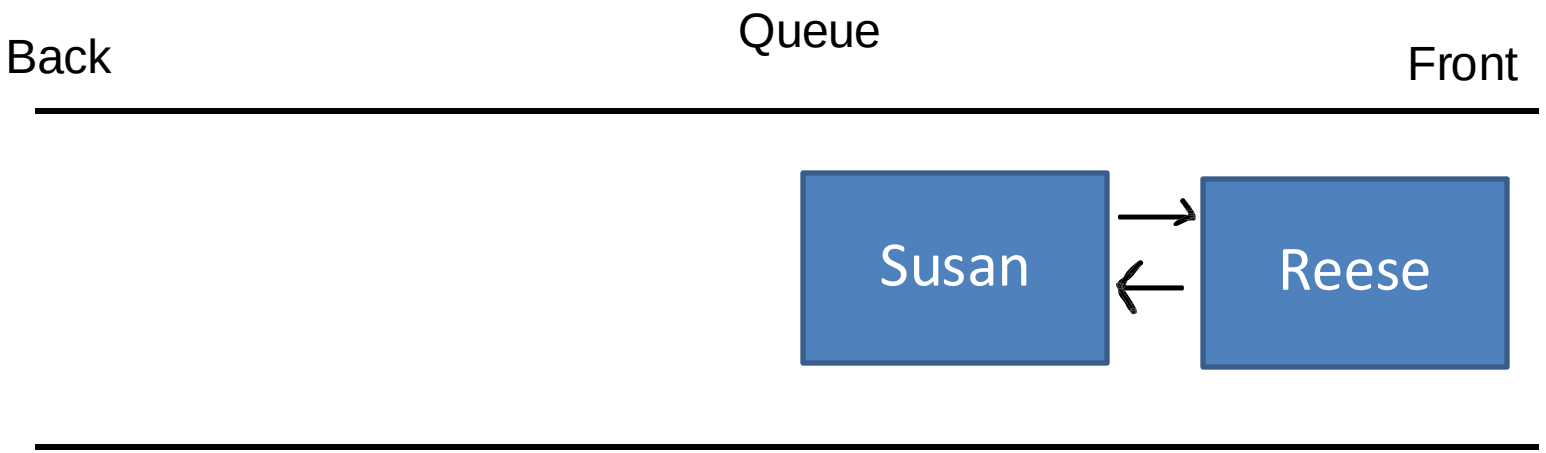
IsEmpty() – Returns true if queue is empty, returns false is queue is not empty

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



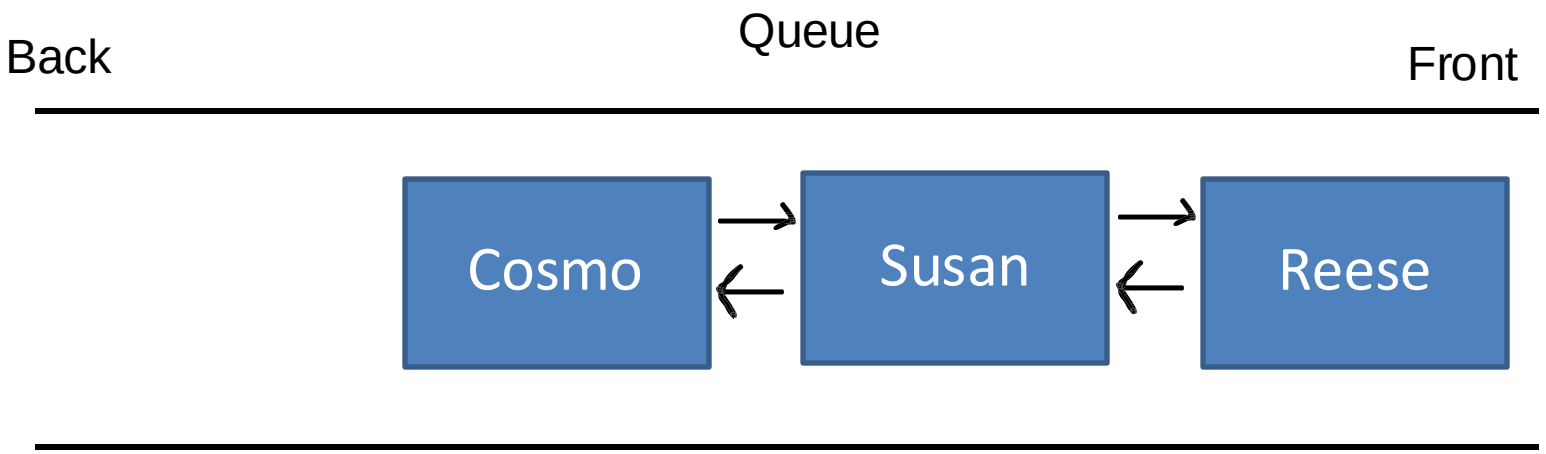
queue.enqueue(“Reese”);

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



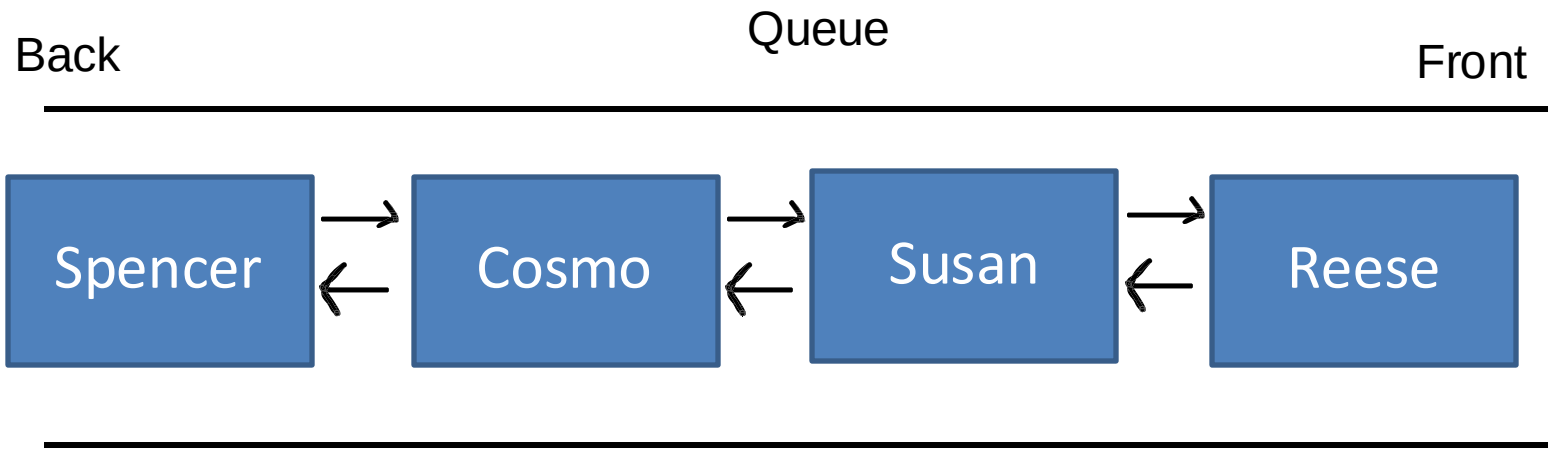

```
queue.enqueue("Reese");
queue.enqueue("Susan");
```

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



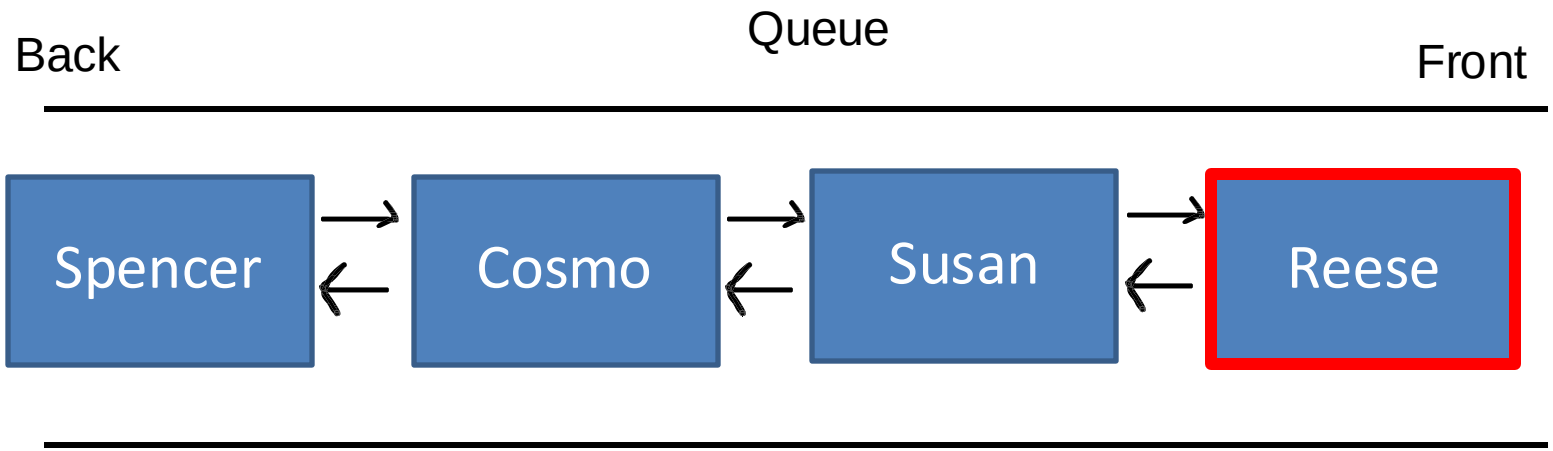

```
queue.enqueue(“Reese”);
queue.enqueue(“Susan”);
queue.enqueue(“Cosmo”);
```

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue(“Reese”);
queue.enqueue(“Susan”);
queue.enqueue(“Cosmo”);
queue.enqueue(“Spencer”);
```

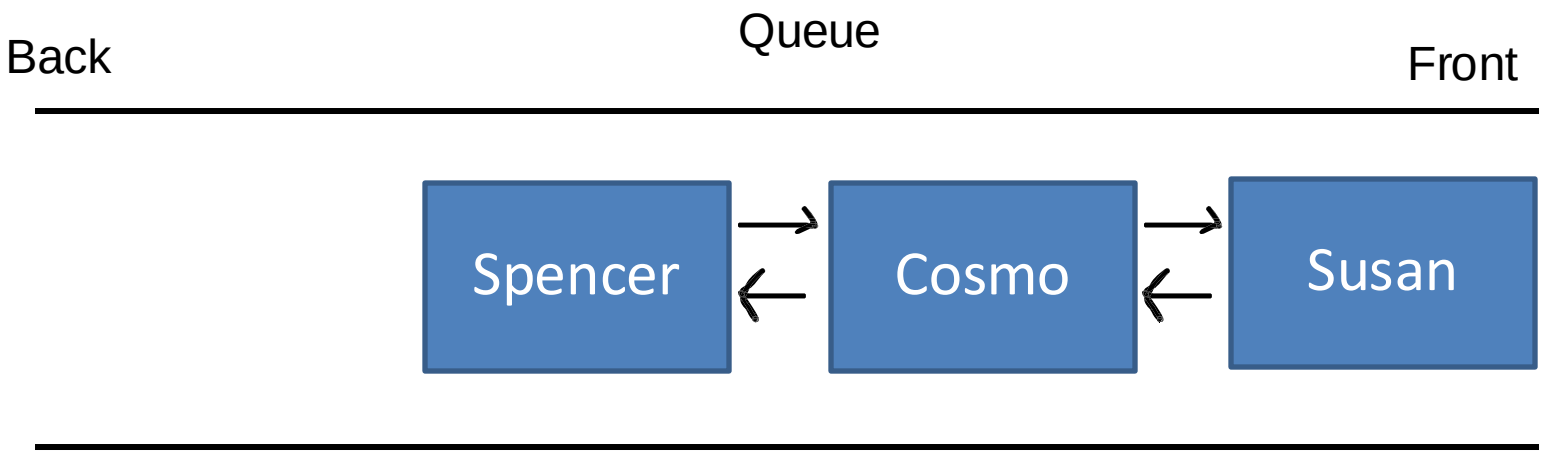
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue("Reese");
queue.enqueue("Susan");
queue.enqueue("Cosmo");
queue.enqueue("Spencer");
```

```
queue.dequeue()
```

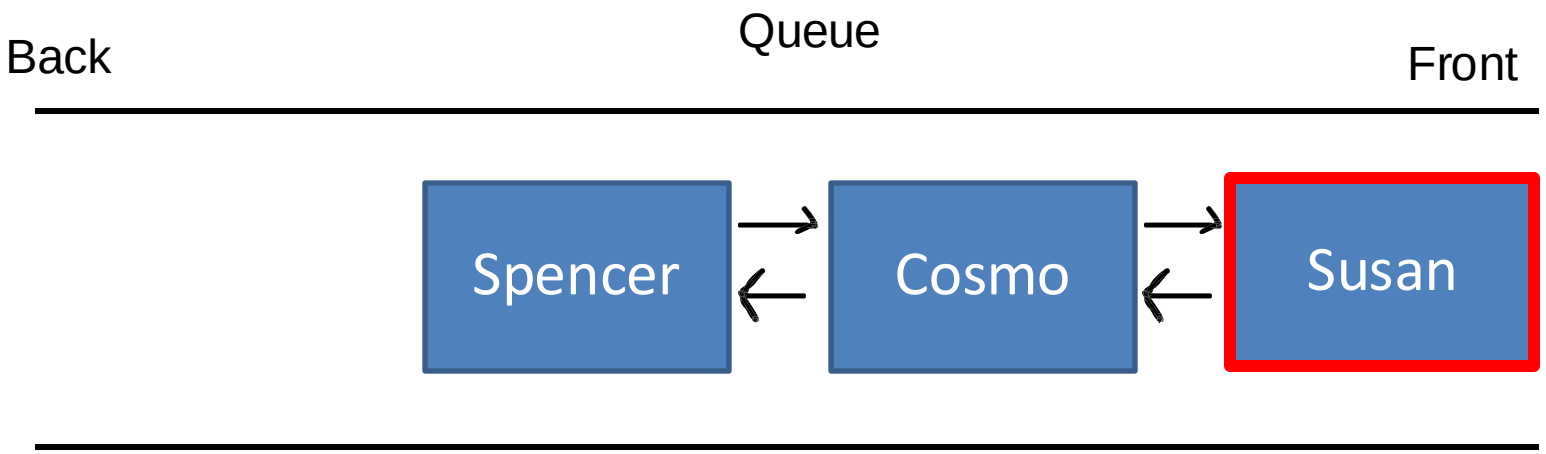
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue("Reese");
queue.enqueue("Susan");
queue.enqueue("Cosmo");
queue.enqueue("Spencer");
```

```
queue. dequeue()
```

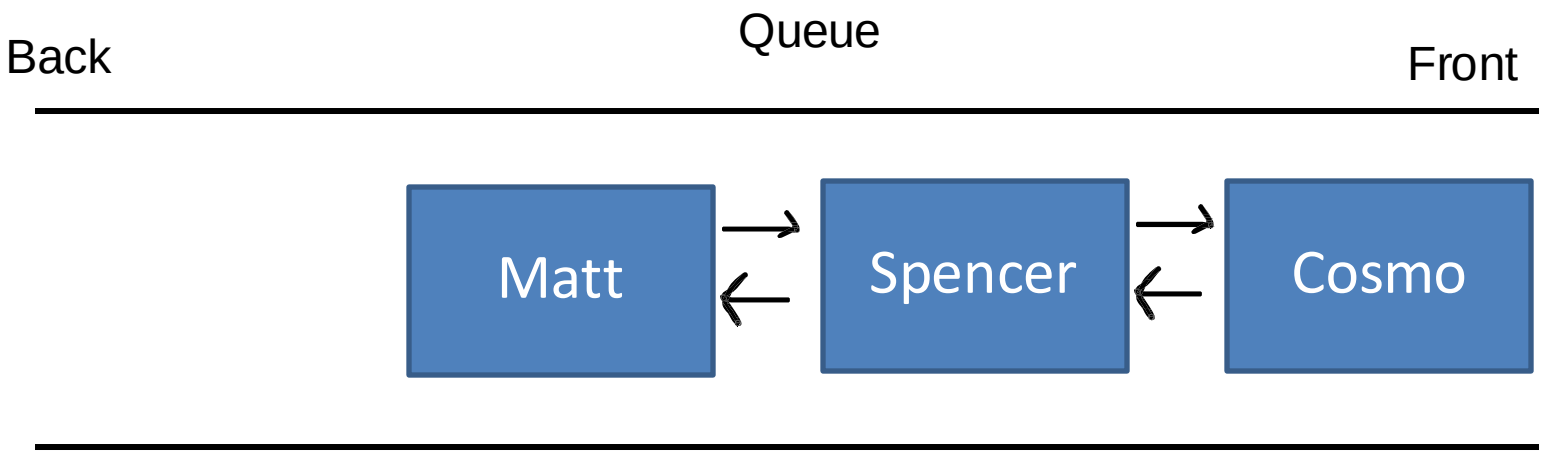
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue("Reese");
queue.enqueue("Susan");
queue.enqueue("Cosmo");
queue.enqueue("Spencer");
```

```
queue. dequeue()
queue. dequeue()
```

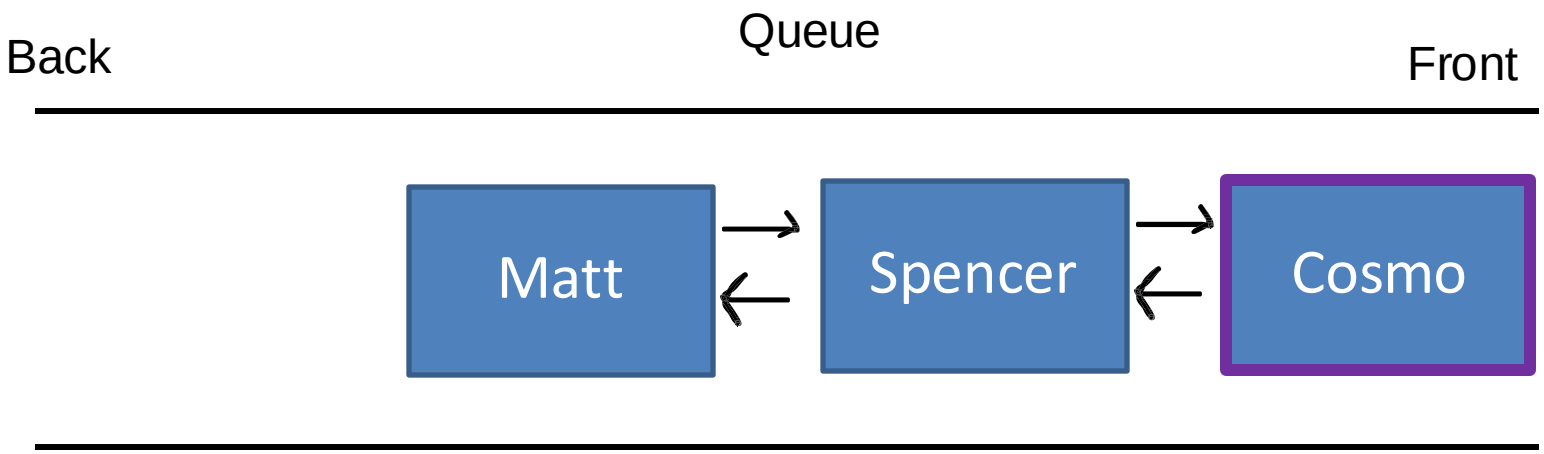
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue("Reese");
queue.enqueue("Susan");
queue.enqueue("Cosmo");
queue.enqueue("Spencer");
```

```
queue. dequeue()
queue. dequeue()

queue.enqueue("Matt");
```

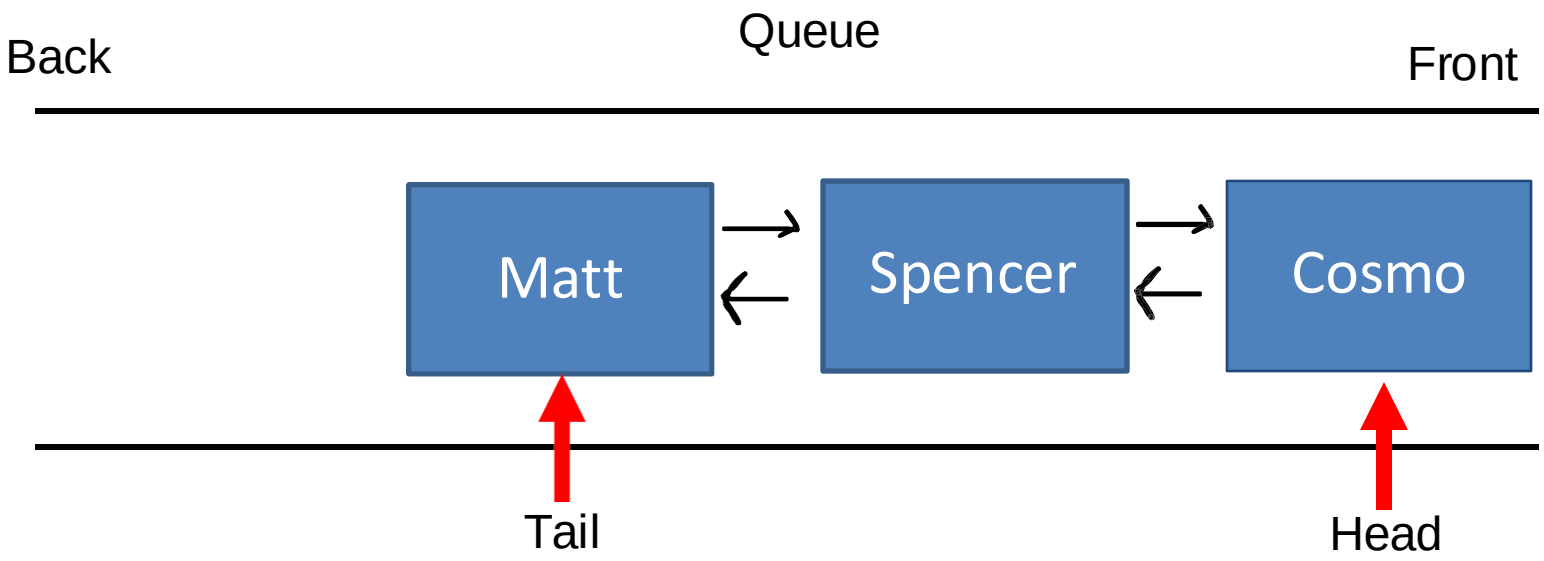
A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion




```
queue.enqueue("Reese");
queue.enqueue("Susan");
queue.enqueue("Cosmo");
queue.enqueue("Spencer");
```

```
queue. dequeue()
queue. dequeue()

queue.enqueue("Matt");
queue.peek()
    → "Cosmo"
```

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion

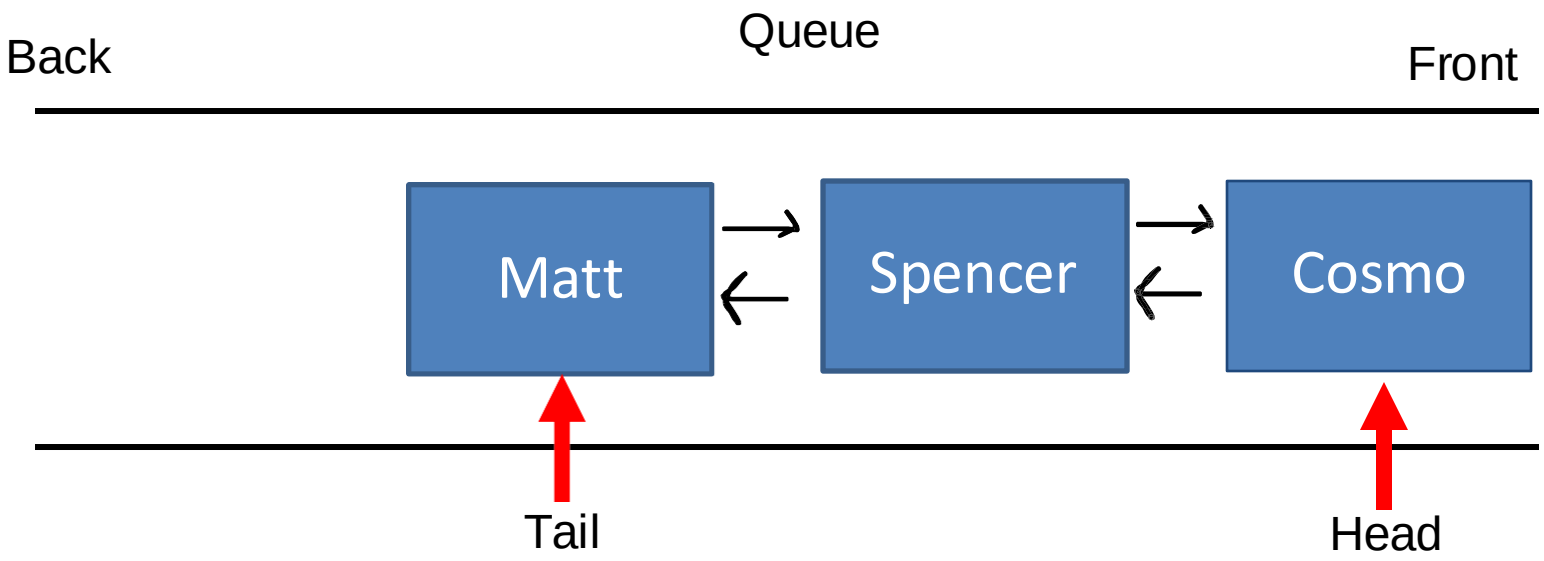


Linked List Implementation

When we enqueue, we add the element to ???

When we dequeue, we remove the element from ???

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion

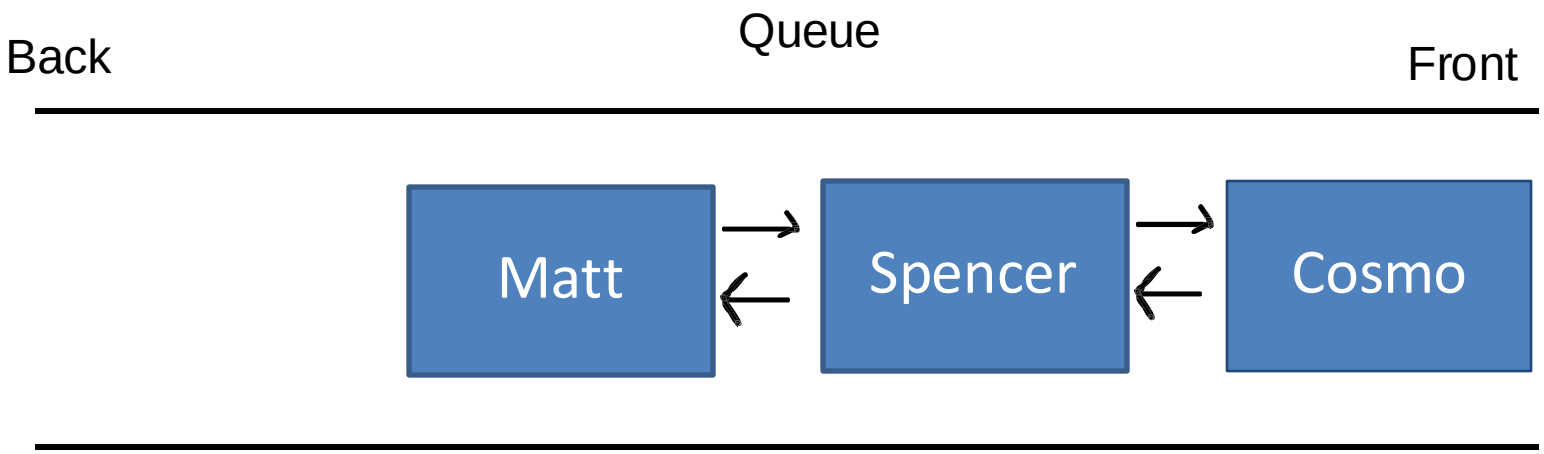


Linked List Implementation

When we enqueue, we add the element to the end of the linked list

When we dequeue, we remove the element from the beginning of the linked list

A **Queue** is a data structure that holds data, but operates in a First-in First-out (**FIFO**) fashion



As we use our queue, we might need to keep track of a few things

- The **size** of the queue
- The **front** of the queue (not when we use LLs)
- The **back** of the queue (not when we use LLs)