Implementing a Queue…

Queue

```
class QueueOfIntegers
class QueueOfFloat
class QueueOfString
…
class QueueOfStackOfIntegers
```

Problem:

- Rewriting code is tedious
- Maintaining many copies leads to errors

Parameterized Data Types

Generics

Parameterize queue with desired type

```
Queue<String> stringQueue = new Queue<String>();
Queue<Job> jobQueue = new Queue<Job>();

String name = “Joe”;
Job work = new Job(“CleanLaundry”);

stringQueue.enqueue(name);
jobQueue.enqueue(work);

Job nextJob = jobQueue.dequeue();
```

Generic Stack

```
public class SpecialStack<ItemType> {

      ItemType[] m_array = new ItemType[MAXITEMS];

      public void push(ItemType key){
         ...
      }

      public ItemType pop(){
         ...
      }
}
```

Parameterized Data Types

Error!

```
Queue<int> intQueue = new Queue<int>();


intQueue.enqueue(23);
int i = intQueue.dequeue();
```

Parameterized Data Types

Wrappers

Error!

```
Queue<int> intQueue = new Queue<int>();


intQueue.enqueue(23);
int i = intQueue.dequeue();
```

Parameterized Data Types

Wrappers

```
Queue<Integer> intQueue = new Queue<Integer>();


intQueue.enqueue(23);
int i = intQueue.dequeue();
```

- Integer wraps int
- Float wraps float
- Character wraps char
- Boolean wraps boolean

Parameterized Data Types

Wrappers

AutoBoxing

```
Queue<Integer> intQueue = new Queue<Integer>();


intQueue.enqueue(23);

int i = intQueue.dequeue();
```

AutoUnboxing