HW01 – Generics
Assigned: Sunday, May 23rd
Due: Friday, May 28th, 2010 before 10:00pm

Objectives
This homework assignment will familiarize you with generics and arrays in java as you utilize them to implement a simple data structure known as a queue.

Requirements
You must use an array data structure to implement a generic queue in **MyQueue.java**. The final version of the generic queue should allow you to instantiate the queue for any object type; for example, I should be able to call **new MyQueue<String>()** to get an empty queue that can hold String values. Your queue should follow the "first in, first out" policy. At any point, the next element to be dequeued should always be the first element in your array.

Your **MyQueue** class must have the following functionality:

- A private array of the generic type that serves as the queue.
  Hint: arrays do not automatically increase in size when they become full so you may need a bigger array than the one you originally created at some point.
- An empty constructor that simply creates an empty queue of the proper generic type.
- A constructor that types in a generic value and creates a queue that initially contains the generic value passed in.
- A public method *enqueue* that takes in a generic value and adds the value to the queue. This method should return nothing.
- A public method *dequeue* that has no parameters and removes the next item to be removed from the queue and returns this item. This method's return type should be the generic type of the queue.
- A public method *purge* that takes in no parameters and purges the queue of all items. This method should return nothing.
- A public method *isEmpty* that takes in nothing and returns a boolean denoting whether the queue is currently empty.
- A public method *getSize* that takes in nothing and returns the number of items currently held in the queue as an int.
- A public method *printQueue* that takes in nothing and returns a String representation of the elements of the queue. Each element, beginning with the next element to be dequeued and ending with the last element to be dequeued, should be printed with a space inserted between their String representations.
  Hint: If I created a queue which held Integer values (new MyQueue<Integer>()), then one possible result of calling printQueue would be "24 78 5 12", where 24 is the next element to be dequeued.
- A *main* method in which both constructors and all of the queue methods are tested thoroughly.

Hint: I recommend instantiating queues of different types and testing all of your methods on each one. Ensure that calling dequeue on an empty queue and enqueue on a full queue function properly.

What to turn in

Turn in all the files you used to write this assignment into t-square. For this assignment, please submit:

- MyQueue.java

By submitting this assignment, you confirm that you understand the CS1332 collaboration policy, and that your submission complies with it.

Keep in mind that you can submit the assignment as many times as you want before the end of the grace period. If you find yourself working close to the time the assignment is due or during the grace period then please submit often since t-square has a habit of going down at the worst possible times.

Remember

These are required, and you will lose points if you do not do any of the following:

- Javadocing and Commenting (each method must be clearly and sufficiently commented and/or Javadoced, even the getters and setters)
- Putting your name at the top of each file (or in the @author section in the Java-doc for each file)