3 Object-Oriented Programming (RKH)

The Java class below defines a queue data structure with a fixed capacity.

```
public class FixedCapacityQueue {
        int[] mData = null;
        int   mHead=0; // index of first full cell
        int   mTail=0; // index of next empty cell

        public FixedCapacityQueue(int capacity) {
                mData = new int[capacity];
        }

        public boolean enqueue(int x) {
                if (mTail==mData.length) return false;
                mData[mTail]=x;
                mTail++;
                return true;
        }

        public int dequeue() {
                if (mTail==mHead) return -1;
                int v=mData[mHead];
                mHead++;
                return v;
        }
}
```

(*a*) Explain the risks posed by the lack of access modifiers specified on the state. Which access modifier is appropriate for these entities? [3 marks]

(*b*) Discuss the choice to signal enqueue and dequeue errors using return values of false and -1, respectively. Suggest a better alternative and modify `enqueue` and `dequeue` to use it. [5 marks]

(*c*) Explain how an enqueue operation on a `FixedCapacityQueue` object could fail with an error even when there are fewer items in the queue than the assigned capacity of the object. Show how to fix this. [3 marks]

(*d*) Rewrite the class to use Java's Generics so that it can be used to represent queues of arbitrary objects (`Integer`s, `String`s, etc). Use the `java.util.ArrayList` class instead of `int[]` for the type of `mData`. [4 marks]

(*e*) Explain why it was necessary to replace the `int[]` type of `mData` in part (*d*), and why `ArrayList` does not suffer from the same issue. [5 marks]