Tutorial Quiz #1 — Solutions

Recall the `ArrayQueue` implementation given in lecture and reproduced below (in part).

```
public class ArrayQueue implements Queue {
    private int numItems; // the number of items in this queue
    private Object[] contents = new Object[100]; // the items in this queue
    // Methods 'enqueue' and 'dequeue' omitted to save space.
    // Returns the first object in this queue.
    public Object head() {
        return contents[0];
    }
    // Indicates whether this queue is empty.
    public boolean isEmpty() { return numItems == 0; }
} // end ArrayQueue
```

Write a class `EmptyQueueException` which will be used to indicate that the queue is empty.
Answer:

```
public class EmptyQueueException extends Exception {}
```

Marking Scheme:

A. 1 mark for "`class EmptyQueueException ... {}`"

B. 1 mark for "`extends Exception`"

The `head()` and `dequeue()` methods are candidates for throwing an `EmptyQueueException`, since they clearly should not work when the queue is empty. Rewrite the `head()` method so that it throws an `EmptyQueueException` if the queue is empty, or returns the first object in the queue otherwise.
Answer:

```
// Returns the first object in this queue if it is not empty
// (throws an EmptyQueueException if it is empty).
public Object head() throws EmptyQueueException {
    if ( isEmpty() )  throw new EmptyQueueException();
    else  return contents[0];
}
```

Marking Scheme:

C. 1 mark for "`throws EmptyQueueException`" in the method header

D. 1 mark for a correct `if` statement to check whether the queue is empty

E. 1 mark for "`throw new EmptyQueueException();`"