3 Object-Oriented Programming (acr31)

(a) You are given the following implementation for an element of a list:

```java
class Element {
    int item;
    Element next;

    Element(int item, Element next) {
        super();
        this.item = item;
        this.next = next;
    }

    @Override
    public String toString() {
        return item + " " + (next == null ? "" : next);
    }
}
```

(i) What does the statement `super()` mean? [1 mark]

(ii) What is the meaning of `this` in the line `this.item = item`? [1 mark]

(iii) What is the purpose of the annotation `@Override`? [2 marks]

(iv) Rewrite the class to be immutable. You may assume that there are no sub-classes of `Element`. [2 marks]

(b) Use the immutable `Element` class to provide an implementation of an immutable class `FuncList` which behaves like an `int` list in ML. Your class should include a constructor for an empty list and methods `head`, `tail` and `cons` based on the following functions in ML. Ensure that your class behaves appropriately when the list is empty. [6 marks]

```
fun head x::_ = x;
fun cons (x,xs) = x::xs;
fun tail _::xs = xs;
```

(c) Another developer changes your implementation to a generic class `FuncList<T>` that can hold values of any type `T`.

(i) This means that `FuncList<T>` is no longer immutable. Explain why and what could be done to remedy this. [2 marks]

(ii) Java prohibits covariance of generic types. Is this restriction necessary in this case? Explain why with an example. [6 marks]