Concurrent Systems and Applications

(a) The following Java interface describes the API for a first-come, first-served (FCFS) mutual exclusion system, designed to work even if threads are interrupted in the enter and exit routines.

```java
interface FCFS {
    public void enter();
    public void exit();
}
```

Sketch a concrete class, `FCFSImpl`, implementing this interface, which does not need to be re-entrant, ensuring that you satisfy the following requirements:

(i) if a thread is interrupted while executing the entry protocol, it should abort its attempt to gain entry and cleanly terminate the call; you may assume that the calling code will not then enter the critical region; 

(ii) the exit protocol should notify a particular thread and not simply call `notifyAll()`.

(b) Object allocation graphs can be used to detect deadlock in a concurrent application.

(i) Give an example of an object allocation graph and explain the meanings of the different components.

(ii) Describe an algorithm which can use an object allocation graph and an object request matrix to determine whether or not deadlock exists.

(iii) Describe how to distinguish between cases in which deadlock has occurred and those in which deadlock is inevitable but is yet to occur.