1995 Paper 10 Question 3

Further Modula-3

Synchronisation of threads in Modula-3 is achieved through the use of mutexes and condition variables. An alternative scheme would be to use Dijkstra semaphores. A semaphore has a hidden value (usually set to 1 initially) and two atomic operations:

> `wait` (sometimes called P) decrements the stored value. If the result is negative, the thread is suspended; otherwise it continues.
>
> `signal` (sometimes called V) increments the value. If there are any other threads suspended while waiting for the semaphore, one of them is allowed to continue.

Write an interface `Semaphore` defining an opaque object type `T` with `init`, `signal` and `wait` methods. [5 marks]

Sketch an implementation of the `Semaphore` module giving a concrete revelation of `T` and implementing appropriate default methods. [10 marks]

Show how the interface and implementation could be extended to derive a sub-type of `T` with an extra method, `try`, which works like `wait` but returns a `BOOLEAN` value instead of blocking. In the normal case, `try` should return `TRUE` but when the thread would have been suspended, the value in the semaphore is left unchanged and it should return `FALSE`. [5 marks]