T-106.5600 Concurrent Programming Examination 14.5.2007 / hsa

1. The following program consisting of three processes p, q and r, each having one await statement:

```
p: await (i >=  4) i = i - 4;

q: await (i >=  3) i = i - 3;

r: await (i <=  2) i = i + 7;
```

(a) For which initial values of i the program will terminate assuming that scheduling is weakly fair? What are the corresponding final values of i ? (b) Give an argumentation to demonstrate your claims.

2. Show that the following algorithm solves the critical section problem for n processes provided that *delay* is long enough:

```
integer gate = 0

process i:
      loop forever
            non-critical section
            loop
p1:               await gate == 0
p2:               gate = i
p3:               delay
p4:         until gate == i
            critical section
p5:         gate = 0
```

3. In the parallel version of the recursive mergesort algorithm one recursive step consist of dividing an array to two halves, sorting them in parallel and then merging them to one. Assume that each recursive call is executed in its own process with the following outline:

```
parallel_mergesort_process :
      p0: receive A
      p1: divide A to halves
      p2: start sorting the halves
      p3: wait for the results
      p4: merge the halves
      p5: return A
```

Describe how to use semaphores in the above process to synchronize it with others in the recursive calling tree. Both starting and ending of the sort procedure must be synchronized. You should discard the details of passing the actual data and results between the processes and sorting and merging operations.

4. Assume that we have solved a synchronization problem using a general monitor with wait() and signal () operations and two specific condition variables a and b corresponding to two different waiting conditions A and B. Describe how to transform the solution to one that uses Java's synchronized classes.

5. Describe different possible ways to use tuplespace to implement the parallel mergesort described in question 3.