Computation structures

Problem-solving lesson 6

Exercises

1. Three producer processes (P_1, P_2 and P_3) write some integers into a buffer of N slots in such a way that a consumer process C can read them. The producers must access the buffer in an orderly fashion : P_1, then P_2, then P_3, then P_1 an so on. Once the buffer is full, the consumer reads the produced numbers and empties the buffer. Give a simple solution to this problem by using semaphores and a shared memory zone containing *only* the buffer and a counter of the number of elements in it.
2. One producer P and two consumers C_1 and C_2 want to communicate through a buffer mechanism. A buffer of N slots is thus placed between the producer and consumer C_1 while a second buffer, of M slots, is placed between the producer and consumer C_2. Each consumer reads the elements in its buffer, except when the buffer is empty. The producer writes each elements into either the first or the second buffer and must be blocked only when the two buffers are full. Give a simple solution to this problem.

3. The following C code offers a solution to the problem of the shared buffer zone between several producers and several consumers (each element is read by only one consumer) :

```
shared int in = 0, out = 0;
shared int buf[N];
semaphore n = 0, e = N, s = 1;
```

```
append(int v) {
  wait(e);
  wait(s);
  buf[in] = v;
  in = (in + 1) % N;
  signal(s);
  signal(n);
}
```

```
int take() {
  int v;
  wait(s);
  wait(n);
  v = buf[out];
  out = (out + 1) % N
  signal(e);
  signal(s);
  return v;
}
```

(a) Is this solution acceptable ? Explain.

(b) If not, what code modifications are required ? Justify.

(c) Considering that there are K producers and L consumers, modify this program in such a way that each consumer is able to read, at its own pace, *all* the produced elements.