

# Chapter 1

# Process Management

## 1.1. Process Scheduling

## 1.2. Process Synchronization

### 1.2.1. Producer-Consumer Problem

```
1- // Shared bounded buffer
2- class Buffer {
3-     int[] items;
4-     int size;
5-     int count;
6-     int in;
7-     int out;
8-
9-     // Constructor
10-    void Buffer(int capacity) {
11-        size = capacity;
12-        items = new int[size];
13-        count = 0;
14-        in = 0;
15-        out = 0;
16-    }
17-
18-    // Add an item to the buffer (producer)
19-    void put(int value) {
20-        while (count == size) {
21-            wait();    // buffer full
22-        }
23-
24-        items[in] = value;
```

```
25-        in = (in + 1) % size;
26-        count = count + 1;
27-
28-        notify();    // wake up consumer
29-    }
30-
31-    // Remove an item from the buffer (consumer)
32-    int get() {
33-        while (count == 0) {
34-            wait();    // buffer empty
35-        }
36-
37-        int value = items[out];
38-        out = (out + 1) % size;
39-        count = count - 1;
40-
41-        notify();    // wake up producer
42-        return value;
43-    }
44- }
45-
46- // Producer thread
47- class Producer {
48-     Buffer buffer;
49-
50-     void Producer(Buffer b) {
51-         buffer = b;
52-     }
53-
54-     void run() {
55-         int x = 0;
56-
57-         while (true) {
58-             buffer.put(x);
59-             println("Produced: " + x);
60-             x = x + 1;
61-         }
62-     }
63- }
64-
65- // Consumer thread
66- class Consumer {
67-     Buffer buffer;
68-
```

```
69 -     void Consumer(Buffer b) {
70 -         buffer = b;
71 -     }
72 -
73 -     void run() {
74 -         while (true) {
75 -             int v = buffer.get();
76 -             println("Consumed: " + v);
77 -         }
78 -     }
79 - }
80 -
81 - // Program entry point
82 - void main() {
83 -     Buffer buf = new Buffer(5);
84 -
85 -     Producer p = new Producer(buf);
86 -     Consumer c = new Consumer(buf);
87 -
88 -     startThread(p);
89 -     startThread(c);
90 - }
```