

Lösung Aufgabe 2 von Übungsblatt 8 (WS 2006/07)

```
import java.applet.Applet;

public class Sys extends Applet{
    private Producer prod;
    private Consumer cons;

    public void start(){
        Buffer buf = new Buffer(3); // lokal
        prod = new Producer(buf);
        cons = new Consumer(buf); // A und B teilen sich denselben Zaehler
        prod.start();
        cons.start();
    }
    public void stop(){
        prod.stop();
        cons.stop();
    }
}

class Producer implements Runnable{
    private Thread thread;
    private Buffer buf;

    Producer(Buffer b){
        buf = b;
    }
    void start(){
        thread = new Thread(this);
        thread.start();
    }
    void stop(){
        thread.interrupt();
    }
    public void run(){
        String alphabet = "abcdefghijklmnopqrstuvwxyz";
        int ai = 0;
        try{
```

```

        while (true){
            Thread.sleep(500); // hier Unterbrechnung moeglich
            Character c = new Character(alphabet.charAt(ai)); // produce
            buf.put(c);
            ai = (ai+1)%alphabet.length();
        }
    } catch (InterruptedException e) {}
}

class Consumer implements Runnable{
    private Thread thread;
    private Buffer buf;

    Consumer(Buffer b){
        buf = b;
    }
    void start(){
        thread = new Thread(this);
        thread.start();
    }
    void stop(){
        thread.interrupt();
    }
    public void run(){
        try{
            while (true){
                Character c = (Character)buf.get();
                Thread.sleep(750); // consume
            }
        } catch (InterruptedException e) {}
    }
}

class Buffer {
    private Object[] buf;
    private int in = 0; // naechste freie Position
    private int out = 0; // Index d. abzugebenden Elements
    private int N; // Puffergroesse
    private int i = 0; // Anzahl Elemente im Puffer

    Buffer(int s){
        N = s;
    }
}

```

```

    buf = new Object[N]; // Feld der Groesse N
}

synchronized void put(Object obj) throws InterruptedException{
    while (i == N) wait();
    buf[in] = obj;
    i++;
    in = (in+1)%N;
    notify();
    System.out.println("put "+obj+" buf: "
+buf[0]+", "+buf[1]+", "+buf[2]+" in: "+in); //
}

synchronized Object get() throws InterruptedException{
    while (i == 0) wait();
    Object obj = buf[out];
    buf[out] = null; // kann weggelassen werden
    i--;
    out = (out+1)%N;
    notify();
    //System.out.println("get "+obj+"\n");
    System.out.println("get "+obj+" buf: "
+buf[0]+", "+buf[1]+", "+buf[2]+" out: "+out); //
    return obj;
}
}

```