Beispiele mit Loops

```
class LoopExample {
    public static void main (String[] args) {
        int j, m;
        System.out.println("Loop 1");
        j = 0;
        while (j<10) {
            j =  j + 1;
            System.out.print("  " + j);
        }
        System.out.println("");

        System.out.println("Loop 2");
        j = 0;
        while (j++ < 10) {
            System.out.print("  " + j);
        }
        System.out.println("");

        System.out.println("Loop 3");
        j = 0;
        while (++j < 10) {
            System.out.print("  " + j);
        }
        System.out.println("");

        System.out.println("Loop 4");
        j = 0;
        m = 0;
        while (j++  < 10) {
            if ( ++m==j)
            System.out.print(" j= " + j + " m = " + m);
        }
        System.out.println("");

        System.out.println("Loop 5");
        j = 0;
        m = 0;
        while (j++  < 10) {
            if ( m++ == j)
            System.out.print(" j= " + j + " m = " + m);
        }
        System.out.println("");

        System.out.println("Loop 6");
        j = 0;
        m = 0;
        while (j < 10) {
            if ( j==(++m) )
                System.out.print(" j= " + j + " m = " + m);
            j++;
        }
        System.out.println("");

        System.out.println("Loop 7");
        j = 0;
        m = 0;
        while (j < 10) {
            if ( m++==j)
                System.out.print(" j= " + j + " m = " + m);
            j++;
        }
        System.out.println("");

    }
}
```

Part4 mit wait() und notifyAll()

```
public class Main {
      public static void main(String[] args) {
            BoundedBuffer buffer = new BoundedBuffer(1);
            new Thread(new Producer(buffer)).start();
            new Thread(new Consumer(buffer)).start();
      }
}

public interface Buffer {
      void write(int i);
      int read();
}

public class Producer implements Runnable {
      private Buffer buffer;

      public Producer(BoundedBuffer buffer) {
            this.buffer = buffer;
      }

      public void run() {
            //produce an increasing series of integers
            int counter = 0;
            while(counter < Integer.MAX_VALUE) {

                        buffer.write(counter);
                        System.out.println("Producer produced: " +
counter);
                        counter++;
            }
      }
}

public class Consumer implements Runnable {
      private Buffer buffer;

      public Consumer(BoundedBuffer buffer) {
```

```
			this.buffer = buffer;
		}

		public void run() {
			//run essentially forever
			while (true) {
					int value = buffer.read();
					System.out.println("\t\t\tConsumer consumed: "
+ value);
					if(value == Integer.MAX_VALUE)
						return;
			}
		}
}

public class BoundedBuffer implements Buffer{
		private int[] arraybuffer;
		public int count;
		public int start;
		public int end;

		public BoundedBuffer(int i) {
			arraybuffer = new int[i];
			count = 0;
			start = 0;
		}

		public synchronized int read() {
			while (count==0) {
				try {
					wait();
				} catch (InterruptedException e) {}
			}
			if (start== arraybuffer.length)
				start=0;
			int tmp = arraybuffer[start++];
			count--;
			notifyAll();
			return tmp;
		}

		public synchronized void write(int i) {
			while (count==arraybuffer.length){
				try {
					wait();
				} catch (InterruptedException e) {}
			}
			if (end == arraybuffer.length)
				end=0;
			arraybuffer[end++]=i;
			count++;
			notifyAll();
		}
}
```