```
public void initialize() {
   // degree distribution to be averaged over many networks
   degree = new int[N];
   numberOfCompletedNetworks = 0; // will draw many networks
   startNetwork();
}

public void addLink(int i, int j, int s) {
   linkFrom[i*m+s] = j;
   node[i]++;
   node[j]++;
   linkNumber += 2; // twice current number of links
}

public void startNetwork() {
   n = 0;
   linkFrom = new int[m*N];
   node = new int[N];
   x = new double[N];
   y = new double[N];
   linkNumber = 0;
   for(int i = 0;i<=m;i++) {
      n++;
      setPosition(i);
   }
   for(int i = 1;i<m+1;i++) {
      for(int j = 0;j<i;j++) {
         addLink(i, j, j);
      }
   }
}

public void setPosition(int i) {
   double r2min = 1000./N;
   // used to insure two nodes are not drawn too close to each other
   boolean ok = true;
   do {
      ok = true;
      x[i] = Math.random()*100;
      y[i] = Math.random()*100;
      int j = 0;
      while(j<i&&ok) {
         double dx = x[i]-x[j];
         double dy = y[i]-y[j];
         double r2 = dx*dx+dy*dy;
         if(r2<r2min) {
            ok = false;
         }
         j++;
      }
   } while(!ok);
}

public int findNode(int i, int s) {
   boolean ok = true;
   int j = 0;
```



```
   do {
      ok = true;
      int k = (int) (1+Math.random()*linkNumber);
      j = -1;
      int sum = 0;
      do {
         j++;
         sum += node[j];
      } while(k>sum);
      for(int r = 0;r<s;r++) {
         if(linkFrom[i*m+r]==j) {
            ok = false;
         }
      }
   } while(!ok);
   return j;
}

public void addNode(int i) {
   n++;
   if(drawPositions) {
      setPosition(i);
   }
   for(int s = 0;s<m;s++) {
      addLink(i, findNode(i, s), s);
   }
}

public void step() {
   if(n<N) {
      addNode(n);
   } else {
      numberOfCompletedNetworks++;
      // accumulate data for degree distribution
      for(int i = 0;i<n;i++) {
         degree[node[i]]++;
      }
      startNetwork();          // start another network
   }
}

public void degreeDistribution(PlotFrame plot) {
   plot.clearData();
   for(int i = 1;i<N;i++) {
      if(degree[i]>0) {
         plot.append(0, Math.log(i), Math.log(degree[i]*1.0/
               (N*numberOfCompletedNetworks)));
      }
   }
}

public void draw(DrawingPanel panel, Graphics g) {
   if(node!=null&&drawPositions) {
      int pxRadius = Math.abs(panel.xToPix(1.0)-panel.xToPix(0));
      int pyRadius = Math.abs(panel.yToPix(1.0)-panel.yToPix(0));
      g.setColor(Color.green);
```