Patient 0 identification

In [12]:
```
library(igraph)
library(ggplot2)
```

In this lab, we will simulate the propagation of an epidemic using random trees. We will first create the code to simulate a random tree, using two models, the uniform attachment model and the preferential attachment model. Then we will create, step by step, the estimator studied in class to detect the source of the epidemic, We will also study, through simulations, the links between random trees and Polya's urns.

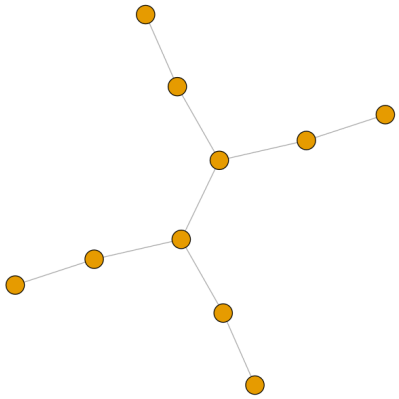
Random tree

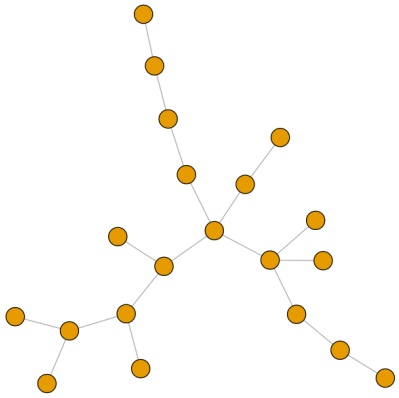
1/ Complete the function *uniform.random.tree*. Remember that each new node selects only one node to connect, *uniformly*. To do so, you can use the function *sample()* from R.

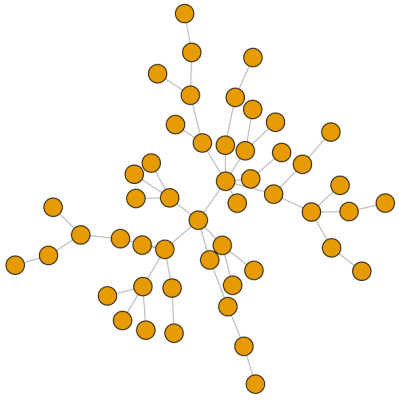
In [13]:
```
uniform.random.tree<-function(Number.of.nodes){

    Adjacency.matrix<-array(0, c(Number.of.nodes,Number.of.nodes))

    for( n in 2:Number.of.nodes){
        node.to.connect <- sample(1:(n-1), 1)
        Adjacency.matrix[node.to.connect, n] <- 1
        Adjacency.matrix[n, node.to.connect] <- 1
    }
    return(Adjacency.matrix)
}
```

2/ Plot the tree generated by the uniform tree function for 10, 20, 50 nodes. The code below can be used as help. What did you check in your plot to be sure that your function is working?

In [14]:
```
node.sizes = c(10, 20, 50)

for(nb.nodes in node.sizes){
    A = uniform.random.tree(nb.nodes)
    g <- graph_from_adjacency_matrix(A,mode = "undirected")
    plot(g,vertex.size=10,layout = layout_with_kk,vertex.label=NA)
}
```







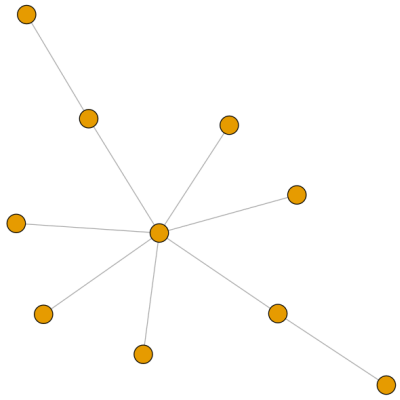
To ensure that the function is working correctly, we can observe graphically that the graphs are effectively trees (no cross connections across branches) with the corresponding number of nodes.

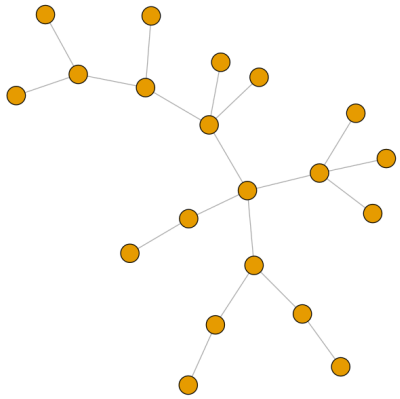
3/ In the preferential attachement model, a new node $n+1$ will decide to connect to node $i \in \{1, \ldots, n\}$ with probability $\frac{d_i}{\sum_{i=1}^{n} d_n}$, where d_i is the degree of node i.

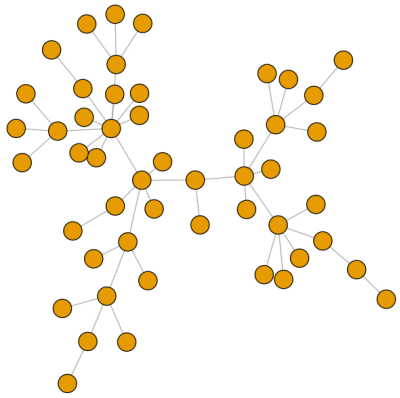
Complete the function *preferential attachement.random.tree* with this selection mechanism. Plot the tree generated by the *preferential.attachement.random.tree* functions for 10, 20, 50 nodes.

In [15]:
```
preferential.attachement.random.tree<-function(Number.of.nodes){

    Adjacency.matrix<-array(0,c(Number.of.nodes,Number.of.nodes))
    Adjacency.matrix[1,2] = 1
    Adjacency.matrix[2,1] = 1

    for( n in 3:Number.of.nodes){
        degrees <- colSums(Adjacency.matrix[1:(n-1), 1:(n-1)])
        probabilities <- degrees / sum(degrees)

        # Select node to attach the new node
        node.to.connect <- sample(1:(n-1), 1, prob = probabilities)

        Adjacency.matrix[node.to.connect, n] <- 1
        Adjacency.matrix[n, node.to.connect] <- 1
    }
    return(Adjacency.matrix)
}
```

In [16]:
```
node.sizes = c(10, 20, 50)

for(nb.nodes in node.sizes){
    A = preferential.attachement.random.tree(nb.nodes)
    g <- graph_from_adjacency_matrix(A,mode = "undirected")
    plot(g,vertex.size=10,layout = layout_with_kk,vertex.label=NA)

}
```



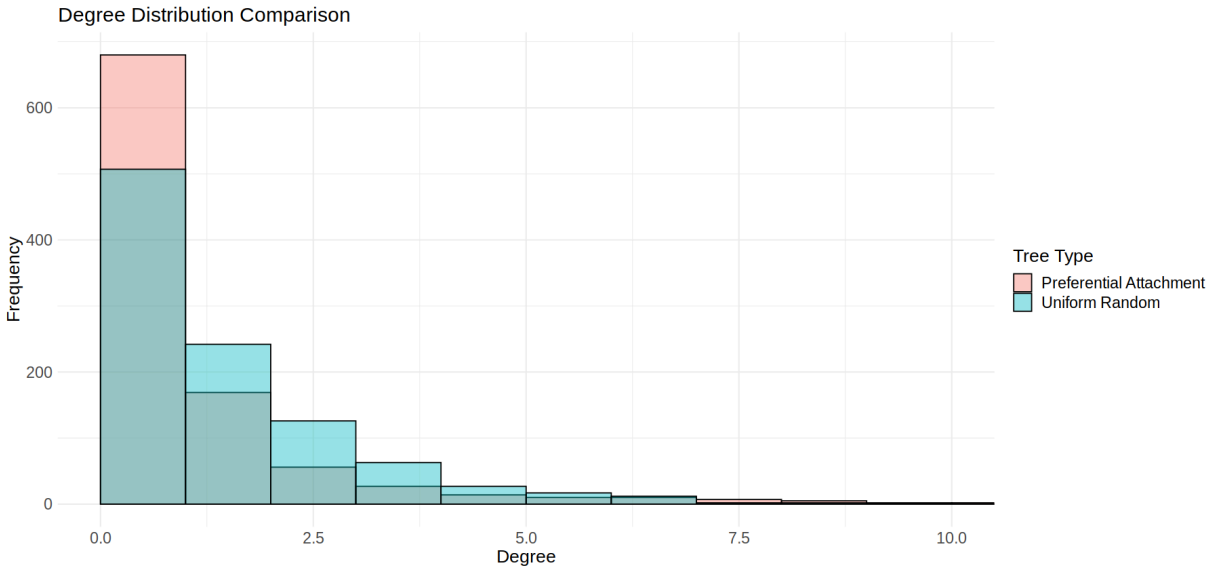




In [17]:
```r
nb.nodes <- 1000

un.A = uniform.random.tree(nb.nodes)
un.degrees <- colSums(un.A)
cat("\nMaximum degree of uniform random tree:", max(un.degrees))

pa.A = preferential.attachement.random.tree(nb.nodes)
pa.degrees <- colSums(pa.A)
cat("\nMaximum degree of preferential attachement random tree:", max(pa.degr

# CHAT GPT from this point
# Create data frames for ggplot
pa_data <- data.frame(NodeType = "Preferential Attachment", Degree = pa.degr
un_data <- data.frame(NodeType = "Uniform Random", Degree = un.degrees)

combined_data <- rbind(pa_data, un_data)

# Set the maximum degree value
deg.max <- 10

# Plot histogram with adjusted figure size, bins, text size, and limited x-a
```

```
options(repr.plot.width = 15)
ggplot(combined_data, aes(x = Degree, fill = NodeType)) +
  geom_histogram(position = "identity", alpha = 0.4, breaks = seq(0, deg.max
  labs(title = "Degree Distribution Comparison",
       x = "Degree", y = "Frequency",
       fill = "Tree Type") +
  theme_minimal() +
  theme(
    text = element_text(size = 16),
    plot.title = element_text(size = 18),
    axis.text = element_text(size = 14),
    axis.title = element_text(size = 16),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 16)
  ) +
  coord_cartesian(xlim = c(0, deg.max))  # Limit x-axis to deg.max
```

```
Maximum degree of uniform random tree: 12
Maximum degree of preferential attachement random tree: 61
```



Implementation of the estimator

5/ Complete the code below which implements the Ψ function described in class (Please call it Psi). You will have to use the function *components* and *groups*. Can you explain the purpose of these two functions?

We recall the definition: Let $T = (V, E)$ be a tree, and $u \in V$. We define $T - \{u\}$ to be the graph obtained by removing u and the links starting from u in T. For $v \in \mathcal{N}(u)$, let $T_v^{(u)}$ be the connexe component of $T - \{u\}$ which is containing v. The function $\Psi_T : V(T) \to \mathbb{N}$ is defined by:

$$\Psi_T(u) = \max_{v \in \mathcal{N}(u)} \mid T_v^{(u)} \mid.$$

In [18]:
```
Psi<-function(u,Adjacency.matrix){

    Adjacency.matrix[u, ] <- 0
    Adjacency.matrix[, u] <- 0

    connected.components <- components(graph_from_adjacency_matrix(Adjacency
    component.sizes <- sapply(groups(connected.components), length) #groups(

    return(max(component.sizes))
}
```

6.1/ Implenent the function $A_K(\cdot)$ seen in class. The argument of the function is K and *Adjacency.matrix*. Test this function for $K = 5, 10$, with the uniform model and the preferential attachement model ($50, 100, 1000$ nodes). Interpret the results.

Definition of $A_K(\cdot)$: $A_K(T)$ is the mapping which returns the set of K vertices with smallest $\Psi_T(u)$ values (ties are broken arbitrarily).

6.2/ Plot the confidence intervals of the estimator using the function *quantile* for the parameters defined in the previous question. To do so, you will have to perform each set-up multiple times (for instance 100 times). Interpret your results.

In [19]:
```
A<-function(K,Adjacency.matrix){
    number.of.nodes<-dim(Adjacency.matrix)[1]
    Psi.vector<-c()

    for(i in 1:number.of.nodes){
        Psi.vector <- c(Psi.vector, Psi(i, Adjacency.matrix))
    }

    df<-data.frame(nodes=order(Psi.vector)[1:K],value=sort(Psi.vector)[1:K])
    return(df)
}
```

In [20]:
```
K.values <- c(5, 10)
nb.nodes.values <- c(50, 100, 1000)


for(nb.nodes in nb.nodes.values){
    for(K in K.values){
        pa.adj = preferential.attachement.random.tree(nb.nodes)
        un.adj = uniform.random.tree(nb.nodes)
        Ak.pa <- A(K, pa.adj)
        Ak.un <- A(K, un.adj)
        cat("\nK: ", K, " Number of nodes: ", nb.nodes, "\n")
        cat("\nUniform Random Tree")
        print(Ak.un[1])
        cat("\nPreferential Aattachement Random Tree")
        print(Ak.pa[1])
    }
}
```

K: 5 Number of nodes: 50

Uniform Random Tree nodes
1   1
2   2
3   3
4   4
5   11

Preferential Aattachement Random Tree nodes
1   1
2   3
3   5
4   2
5   22

K: 10 Number of nodes: 50

Uniform Random Tree   nodes
1   1
2   3
3   10
4   9
5   4
6   19
7   7
8   6
9   14
10   29

Preferential Aattachement Random Tree   nodes
1   1
2   4
3   7
4   10
5   2
6   13
7   12
8   5
9   8
10   14

K: 5 Number of nodes: 100

Uniform Random Tree nodes
1   2
2   1
3   12
4   5
5   6

Preferential Aattachement Random Tree nodes
1   2
2   3
3   4
4   5

```
5     6

K:  10  Number of nodes:  100

Uniform Random Tree   nodes
1      5
2      1
3      2
4      8
5     15
6      6
7     20
8     25
9     31
10     3

Preferential Aattachement Random Tree   nodes
1      1
2      7
3      3
4     12
5     13
6     21
7      4
8     41
9     10
10    28

K:  5  Number of nodes:  1000

Uniform Random Tree  nodes
1     1
2     2
3     7
4     3
5    16

Preferential Aattachement Random Tree  nodes
1     1
2     5
3    10
4     2
5     6

K:  10  Number of nodes:  1000

Uniform Random Tree   nodes
1      1
2      2
3      3
4      5
5      9
6      4
7      8
8      7
9     11
```

```
10    27

Preferential Aattachement Random Tree   nodes
1      2
2      3
3      1
4      4
5      7
6      5
7     49
8     74
9     79
10     6
```

As can be observed, the patient 0 is usually correctly identified inside the group $A(k)$. However, some exceptions can be found. To evaluate the performance of the detection, confidence intervals can be plotted.

In [21]:
```r
Patient.zero.order <- function(Adjacency.matrix){
    number.of.nodes<-dim(Adjacency.matrix)[1]
    Psi.vector<-c()

    for(i in 1:number.of.nodes){
        Psi.vector <- c(Psi.vector, Psi(i, Adjacency.matrix))
    }
    df <- data.frame(nodes = order(Psi.vector), value = sort(Psi.vector))
    return(which(df$nodes == 1))
}
```

In [23]:
```r
nb.nodes.values <- c(50, 100)
# 1000 nodes would require a very long simulation time, around 1hr for 100 s
num_simulations <- 1000  # Number of simulations

results_PA <- c()
results_UN <- c()

for (nb.nodes in nb.nodes.values) {
    for (sim in 1:num_simulations) {
        pa.adj <- preferential.attachement.random.tree(nb.nodes)
        un.adj <- uniform.random.tree(nb.nodes)
        results_PA <- c(results_PA, Patient.zero.order(pa.adj))
        results_UN <- c(results_UN, Patient.zero.order(un.adj))
    }
    cat("\nNumber of nodes: ", nb.nodes, "\n")
    cat("\nUniform Random Tree")
    cat("\n25% quantile =", quantile(results_UN, 0.25))
    cat(", 75% quantile =", quantile(results_UN, 0.75))
    cat(", 90% quantile =", quantile(results_UN, 0.90), "\n")

    cat("\nPreferential Attachement Random Tree")
    cat("\n25% quantile =", quantile(results_PA, 0.25))
    cat(", 75% quantile =", quantile(results_PA, 0.75))
    cat(", 90% quantile =", quantile(results_PA, 0.90), "\n")

    results_PA <- c()
```

```
    results_UN <- c()
}
```

```
Number of nodes:  50

Uniform Random Tree
25% quantile = 1, 75% quantile = 4, 90% quantile = 9

Preferential Attachement Random Tree
25% quantile = 1, 75% quantile = 4, 90% quantile = 13

Number of nodes:  100

Uniform Random Tree
25% quantile = 1, 75% quantile = 5, 90% quantile = 9

Preferential Attachement Random Tree
25% quantile = 1, 75% quantile = 5, 90% quantile = 15
```

The easiest conclusion is that as there are more nodes, the confidence intervals grow, meaning that k has to be higher in $A(k)$ so that more nodes are considered as possible patient zero candidates.

A more complex situation takes place when comparing preferential attachement and uniform random trees. In all cases, it seems that patient zero prediction is easier in Uniform Random Trees. A possible explanation for this is that Uniform Trees are less dense near the center of the graph.

7/ (Bonus) Create your own estimator of the culprit. Test it on the different models and interpret your results.

In [ ]: