A+ Computer Science **Graph – CONNECTIONS 1** **Lab Value - 100**

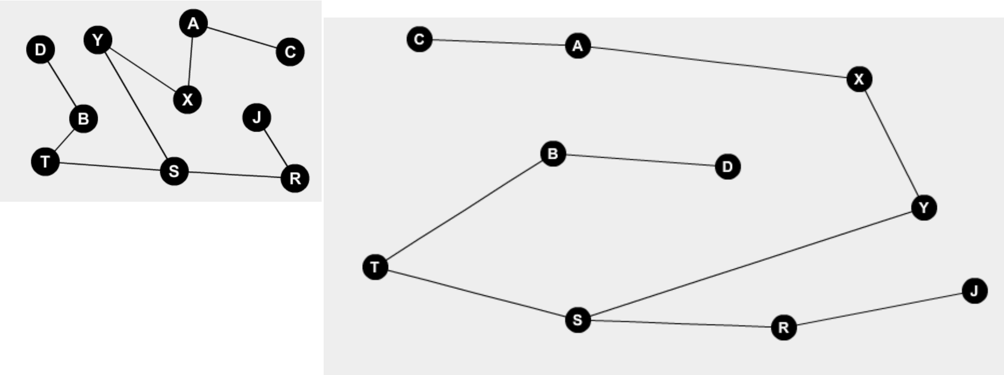
Lab Goal : This lab was designed to teach you how to use recursion to solve a connection problem.

Lab Description : Take a group of provided connections and create a map of all connections. Search this map with two provided values to see if a connection exists. All connections are bi-directional.

If A is connected to B and B is connected to C, is A connected to C? yes

Example 1 (this is one tree drawn two different ways):

```
CA XY RS YS ST TB AX BD RJ
CD

C connects to D == yes
```



Sample Data :

```
9
CA XY RS YS ST TB AX BD RJ
CD
PQ QX AX BX CX DX EX FX GX AB BC CD DE AE CE FD TG
PT
AE EI IO OU BC CD DF FG
AG
HI HJ HK KL KM KN MO MP MQ
HQ
AB CD EF GH CB ED GF HI
AI
TV XY AZ XT JK KL LT JX MN TN JL NO OP PT NX
VZ
AB BC CD DE EF FG GH HI IJ JA AC FZ
AZ
NO PQ RS TU OU RP AB CD EF GH AH CE NS FA GQ
DT
IX VX CX DX MX LX BY
IB
```

Files Needed ::

Graph.java
GraphRunner.java
graph1.dat

Sample Output :

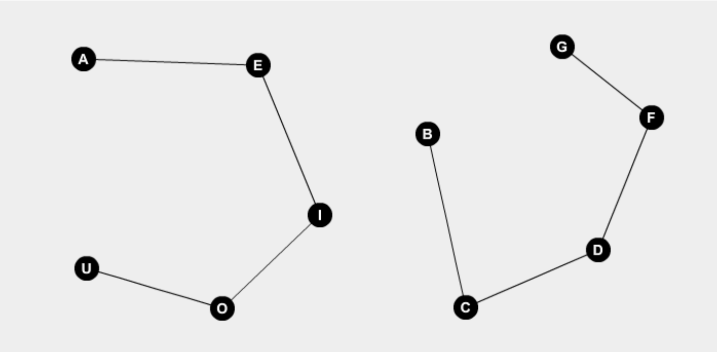
```
C connects to D == yes
P connects to T == yes
A connects to G == no
H connects to Q == yes
A connects to I == yes
V connects to Z == no
A connects to Z == yes
D connects to T == yes
I connects to B == no
```

algorithm help

```
check(String one, String two, String been)
{
   if a direct connection exists between one and two
      shut it down – we have a match
   else
   {
       get the current list of connections for one
       loop through all of the connections
       If you have not checked the current spot
            add current spot to been
            check to see a connection exists between spot and the destination ( recursive call )
   }
}
```

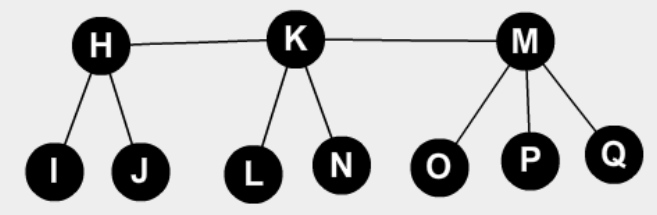
Example 2 (this graph is a forest comprised of two trees.)

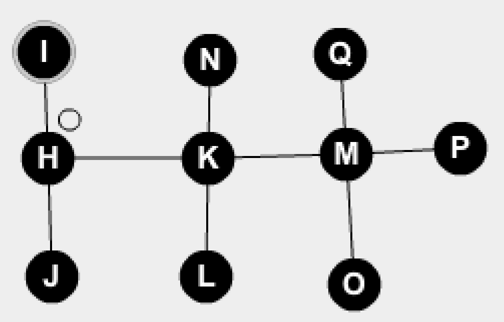
```
AE EI IO OU BC CD DF FG
AG

A connects to G == no
```



Example 3 (this is one tree)

```
Given:  HI HJ HK KL KM KN MO MP MQ

The map is:
H  IJK
I  H
J  H
K  HLMN
L  K
M  KOPQ
N  K
O  M
P  M
Q  M
```





H connects to Q == yes

Call stack:

```
connected(H, Q, H)     = connected(I, Q, HH) | connected(J, Q, HH) | connected(K, Q, HH)

connected(I, Q, HH)    = false

connected(J, Q, HH)    = false

connected(K, Q, HH)    = connected(L, Q, HHK) | connected(M, Q, HHK) | connected(N, Q, HHK)

connected(L, Q, HHK)   = false

connected(M, Q, HHK)   = connected(O, Q, HHKM) | connected(P, Q, HHKM) | connected(Q, Q, HHKM)

connected(N, Q, HHK)   = false

connected(O, Q, HHKM)  = false

connected(P, Q, HHKM)  = false

connected(Q, Q, HHKM)  = true
```