NAME: VINAY REDDY MALLIDI REG.NO :15BCE0687

SLOT: A1+TA1

DIGITAL ASSESSMENT-2

SOCIAL AND INFORMATION NETWORKS

A symmetrized snapshot of the structure of the Internet at the level of Autonomous system (AS) – Given real world social network.

Use R programming

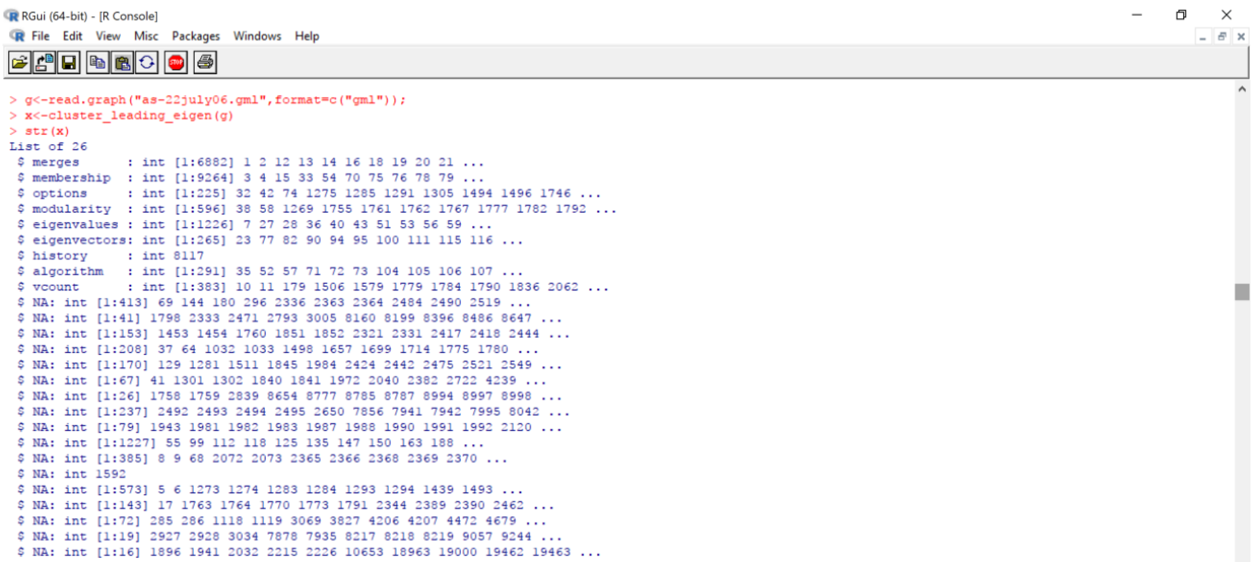
1) **Find the number of Autonoumous systems in the network and top 20 popular Ases. Also give the details of the organizations and countries to which top 20 ASes belong.**

CODE:

```
> g<-read.graph("as-22july06.gml",format=c("gml"));
> x<-cluster_leading_eigen(g)
> str(x)
List of 26
 $ merges     : int [1:6882] 1 2 12 13 14 16 18 19 20 21 ...
 $ membership  : int [1:9264] 3 4 15 33 54 70 75 76 78 79 ...
 $ options     : int [1:225] 32 42 74 1275 1285 1291 1305 1494 1496 1746 ...
 $ modularity  : int [1:596] 38 58 1269 1755 1761 1762 1767 1777 1782 1792 ...
 $ eigenvalues : int [1:1226] 7 27 28 36 40 43 51 53 56 59 ...
 $ eigenvectors: int [1:265] 23 77 82 90 94 95 100 111 115 116 ...
 $ history     : int 8117
 $ algorithm   : int [1:291] 35 52 57 71 72 73 104 105 106 107 ...
 $ vcount      : int [1:383] 10 11 179 1506 1579 1779 1784 1790 1836 2062 ...
 $ NA: int [1:413] 69 144 180 296 2336 2363 2364 2484 2490 2519 ...
 $ NA: int [1:41] 1798 2333 2471 2793 3005 8160 8199 8396 8486 8647 ...
 $ NA: int [1:153] 1453 1454 1760 1851 1852 2321 2331 2417 2418 2444 ...
 $ NA: int [1:208] 37 64 1032 1033 1498 1657 1699 1714 1775 1780 ...
 $ NA: int [1:170] 129 1281 1511 1845 1984 2424 2442 2475 2521 2549 ...
 $ NA: int [1:67] 41 1301 1302 1840 1841 1972 2040 2382 2722 4239 ...
 $ NA: int [1:26] 1758 1759 2839 8654 8777 8785 8787 8994 8997 8998 ...
 $ NA: int [1:237] 2492 2493 2494 2495 2650 7856 7941 7942 7995 8042 ...
 $ NA: int [1:79] 1943 1981 1982 1983 1987 1988 1990 1991 1992 2120 ...
 $ NA: int [1:1227] 55 99 112 118 125 135 147 150 163 188 ...
 $ NA: int [1:385] 8 9 68 2072 2073 2365 2366 2368 2369 2370 ...
 $ NA: int 1592
 $ NA: int [1:573] 5 6 1273 1274 1283 1284 1293 1294 1439 1493 ...
 $ NA: int [1:143] 17 1763 1764 1770 1773 1791 2344 2389 2390 2462 ...
 $ NA: int [1:72] 285 286 1118 1119 3069 3827 4206 4207 4472 4679 ...
```

$ NA: int [1:19] 2927 2928 3034 7878 7935 8217 8218 8219 9057 9244 ...
$ NA: int [1:16] 1896 1941 2032 2215 2226 10653 18963 19000 19462 19463 ...
- attr(*, "class")= chr "communities"

OUTPUT:



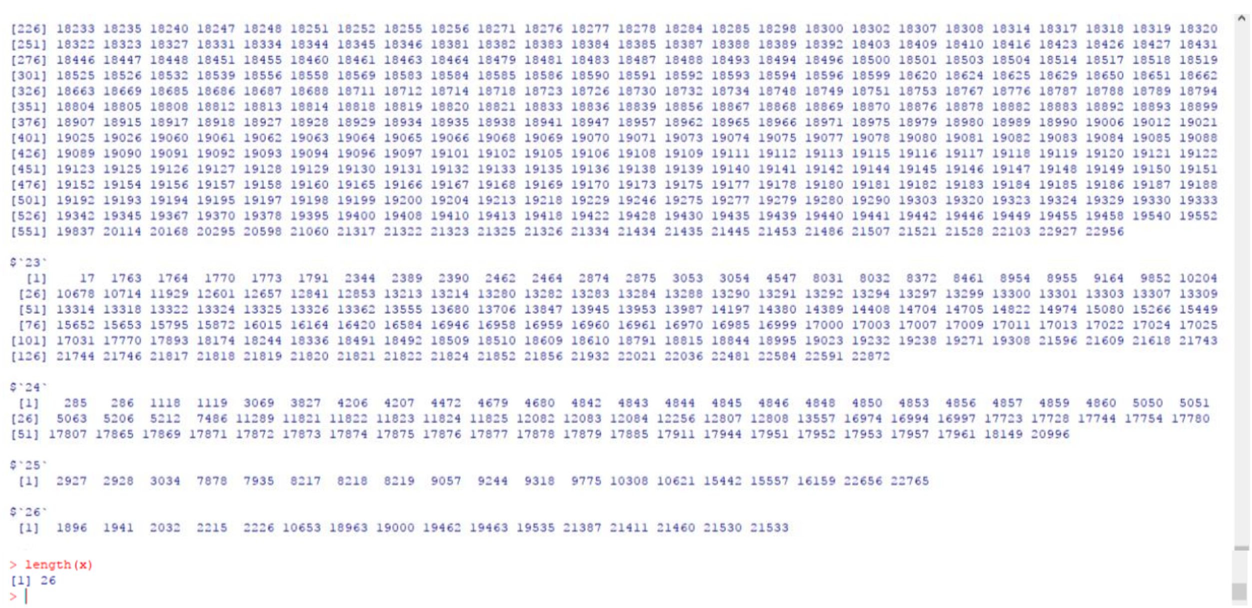
```
RGui (64-bit) - [R Console]
File  Edit  View  Misc  Packages  Windows  Help

> g<-read.graph("as-22july06.gml",format=c("gml"));
> x<-cluster_leading_eigen(g)
> str(x)
List of 26
 $ merges      : int [1:6882] 1 2 12 13 14 16 18 19 20 21 ...
 $ membership  : int [1:9264] 3 4 15 33 54 70 75 76 78 79 ...
 $ options     : int [1:225] 32 42 74 1275 1285 1291 1305 1494 1496 1746 ...
 $ modularity  : int [1:596] 38 58 1269 1755 1761 1762 1767 1777 1782 1792 ...
 $ eigenvalues : int [1:1226] 7 27 28 36 40 43 51 53 56 59 ...
 $ eigenvectors: int [1:265] 23 77 82 90 94 95 100 111 115 116 ...
 $ history     : int 8117
 $ algorithm   : int [1:291] 35 52 57 71 72 73 104 105 106 107 ...
 $ vcount      : int [1:383] 10 11 179 1506 1579 1779 1784 1790 1836 2062 ...
 $ NA: int [1:413] 69 144 180 296 2336 2363 2364 2484 2490 2519 ...
 $ NA: int [1:41] 1798 2333 2471 2793 3005 8160 8199 8396 8486 8647 ...
 $ NA: int [1:153] 1453 1454 1760 1851 1852 2321 2331 2417 2418 2444 ...
 $ NA: int [1:208] 37 64 1032 1033 1498 1657 1699 1714 1775 1780 ...
 $ NA: int [1:170] 129 1281 1511 1845 1984 2424 2442 2475 2521 2549 ...
 $ NA: int [1:67] 41 1301 1302 1840 1841 1972 2040 2382 2722 4239 ...
 $ NA: int [1:26] 1758 1759 2839 8654 8777 8785 8787 8994 8997 8998 ...
 $ NA: int [1:237] 2492 2493 2494 2495 2650 7856 7941 7942 7995 8042 ...
 $ NA: int [1:79] 1943 1981 1982 1983 1987 1988 1990 1991 1992 2120 ...
 $ NA: int [1:1227] 55 99 112 118 125 135 147 150 163 188 ...
 $ NA: int [1:385] 8 9 68 2072 2073 2365 2366 2368 2369 2370 ...
 $ NA: int 1592
 $ NA: int [1:573] 5 6 1273 1274 1283 1284 1293 1294 1439 1493 ...
 $ NA: int [1:143] 17 1763 1764 1770 1773 1791 2344 2389 2390 2462 ...
 $ NA: int [1:72] 285 286 1118 1119 3069 3827 4206 4207 4472 4679 ...
 $ NA: int [1:19] 2927 2928 3034 7878 7935 8217 8218 8219 9057 9244 ...
 $ NA: int [1:16] 1896 1941 2032 2215 2226 10653 18963 19000 19462 19463 ...
```

2) Find the number of AS links in the network.

CODE:

> g<-read.graph("as-22july06.gml",format=c("gml"));
> x<-cluster_leading_eigen(g)
> str(x)
> x[1:length(x)]
> length(x)

OUTPUT:



```
[226] 18233 18235 18240 18247 18248 18251 18252 18255 18256 18271 18276 18277 18278 18284 18285 18298 18300 18302 18307 18308 18314 18317 18318 18319 18320
[251] 18322 18323 18327 18331 18334 18344 18345 18346 18381 18382 18383 18384 18385 18387 18388 18389 18392 18403 18409 18410 18416 18423 18426 18427 18431
[276] 18446 18447 18448 18451 18455 18460 18461 18463 18464 18479 18481 18483 18487 18488 18493 18494 18496 18500 18501 18503 18504 18514 18517 18518 18519
[301] 18525 18526 18532 18539 18556 18558 18569 18583 18584 18585 18586 18590 18591 18592 18593 18594 18596 18599 18620 18624 18625 18629 18650 18651 18662
[326] 18663 18669 18685 18686 18687 18688 18711 18712 18714 18718 18723 18726 18730 18732 18734 18748 18749 18751 18753 18767 18776 18787 18788 18789 18794
[351] 18804 18805 18808 18812 18813 18814 18818 18819 18820 18821 18833 18836 18839 18856 18867 18868 18869 18870 18876 18878 18882 18883 18892 18893 18899
[376] 18907 18915 18917 18918 18927 18928 18929 18934 18935 18938 18941 18947 18957 18962 18965 18966 18971 18975 18979 18980 18989 18990 19006 19012 19021
[401] 19025 19026 19060 19061 19062 19063 19064 19065 19066 19068 19069 19070 19071 19073 19074 19075 19077 19078 19080 19081 19082 19083 19084 19085 19088
[426] 19089 19090 19091 19092 19093 19094 19096 19097 19101 19102 19105 19106 19108 19109 19111 19112 19113 19115 19116 19117 19118 19119 19120 19121 19122
[451] 19123 19125 19126 19127 19128 19129 19130 19131 19132 19133 19135 19136 19138 19139 19140 19141 19142 19144 19145 19146 19147 19148 19149 19150 19151
[476] 19152 19154 19156 19157 19158 19160 19165 19166 19167 19168 19169 19170 19173 19175 19177 19178 19180 19181 19182 19183 19184 19185 19186 19187 19188
[501] 19192 19193 19194 19195 19197 19198 19199 19200 19204 19213 19218 19229 19246 19275 19277 19279 19280 19290 19303 19320 19323 19324 19329 19330 19333
[526] 19342 19345 19367 19370 19378 19395 19400 19408 19410 19413 19418 19422 19428 19430 19435 19439 19440 19441 19442 19446 19449 19455 19458 19540 19552
[551] 19837 20114 20168 20295 20598 21060 21317 21322 21323 21325 21326 21334 21434 21435 21445 21453 21486 21507 21521 21528 22103 22927 22956

$`23`
  [1]    17  1763  1764  1770  1773  1791  2344  2389  2390  2462  2464  2874  2875  3053  3054  4547  8031  8032  8372  8461  8954  8955  9164  9852 10204
 [26] 10678 10714 11929 12601 12657 12841 12853 13213 13214 13280 13282 13283 13284 13288 13290 13291 13292 13294 13297 13299 13300 13301 13303 13307 13309
 [51] 13314 13318 13322 13324 13325 13326 13362 13555 13680 13706 13847 13945 13953 13987 14197 14380 14389 14408 14704 14705 14822 14974 15080 15266 15449
 [76] 15652 15653 15795 15872 16015 16164 16420 16584 16946 16958 16959 16960 16961 16970 16985 16999 17000 17003 17007 17009 17011 17013 17022 17024 17025
[101] 17031 17770 17893 18174 18244 18336 18491 18492 18509 18510 18609 18610 18791 18815 18844 18995 19023 19232 19238 19271 19308 21596 21609 21618 21743
[126] 21744 21746 21817 21818 21819 21820 21821 21822 21824 21852 21856 21932 22021 22036 22481 22584 22591 22872

$`24`
 [1]   285   286  1118  1119  3069  3827  4206  4207  4472  4679  4680  4842  4843  4844  4845  4846  4848  4850  4853  4856  4857  4859  4860  5050  5051
[26]  5063  5206  5212  7486 11289 11821 11822 11823 11824 11825 12082 12083 12084 12256 12807 12808 13557 16974 16994 16997 17723 17728 17744 17754 17780
[51] 17807 17865 17869 17871 17872 17873 17874 17875 17876 17877 17878 17879 17885 17911 17944 17951 17952 17953 17957 17961 18149 20996

$`25`
 [1]  2927  2928  3034  7878  7935  8217  8218  8219  9057  9244  9318  9775 10308 10621 15442 15557 16159 22656 22765

$`26`
 [1]  1896  1941  2032  2215  2226 10653 18963 19000 19462 19463 19535 21387 21411 21460 21530 21533

> length(x)
[1] 26
> |
```

3) Compute average degree distribution, clustering coefficient and average path length for the given network.

CODE:

```
> g<-read.graph("as-22july06.gml",format=c("gml"))
> vcount(g)
[1] 22963
> ecount(g)
[1] 48436
> b<-average.path.length(g)
> b
[1] 3.842426
> cluster_coeff<-transitivity(g, type = c("undirected", "global", "globalundirected", "localundirected", "local", "average", "localaverage","localaverageundirected", "barrat", "weighted"), vids = NULL,weights = NULL, isolates = c("NaN", "zero"))
> cluster_coeff
[1] 0.01114638
> avg_degree_dist<-mean(degree_distribution(g, cumulative = FALSE))
> avg_degree_dist
[1] 0.000418235
>
```

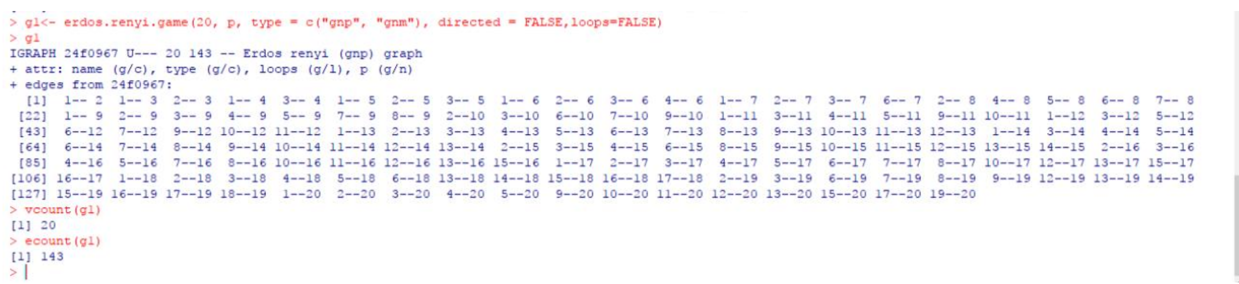
OUTPUT:

4) Simulate the given social network using random graph model G(20,p). Compute the p value from c = (n-1)p, where c is average degree of the given network.

CODE:

> g1<- erdos.renyi.game(20, p, type = c("gnp", "gnm"), directed = FALSE,loops=FALSE)
> g1
IGRAPH 24f0967 U--- 20 143 -- Erdos renyi (gnp) graph
+ attr: name (g/c), type (g/c), loops (g/l), p (g/n)
+ edges from 24f0967:
 [1] 1-- 2 1-- 3 2-- 3 1-- 4 3-- 4 1-- 5 2-- 5 3-- 5 1-- 6 2-- 6 3-- 6 4-- 6 1-- 7 2-- 7 3-- 7 6-- 7 2-- 8 4--
8 5-- 8 6-- 8 7-- 8
 [22] 1-- 9 2-- 9 3-- 9 4-- 9 5-- 9 7-- 9 8-- 9 2--10 3--10 6--10 7--10 9--10 1--11 3--11 4--11 5--11 9-
-11 10--11 1--12 3--12 5--12
 [43] 6--12 7--12 9--12 10--12 11--12 1--13 2--13 3--13 4--13 5--13 6--13 7--13 8--13 9--13 10--13
11--13 12--13 1--14 3--14 4--14 5--14
 [64] 6--14 7--14 8--14 9--14 10--14 11--14 12--14 13--14 2--15 3--15 4--15 6--15 8--15 9--15 10--15
11--15 12--15 13--15 14--15 2--16 3--16
 [85] 4--16 5--16 7--16 8--16 10--16 11--16 12--16 13--16 15--16 1--17 2--17 3--17 4--17 5--17 6--17
7--17 8--17 10--17 12--17 13--17 15--17
[106] 16--17 1--18 2--18 3--18 4--18 5--18 6--18 13--18 14--18 15--18 16--18 17--18 2--19 3--19 6--
19 7--19 8--19 9--19 12--19 13--19 14--19
[127] 15--19 16--19 17--19 18--19 1--20 2--20 3--20 4--20 5--20 9--20 10--20 11--20 12--20 13--20 15--
20 17--20 19--20
> vcount(g1)
[1] 20
> ecount(g1)
[1] 143
>

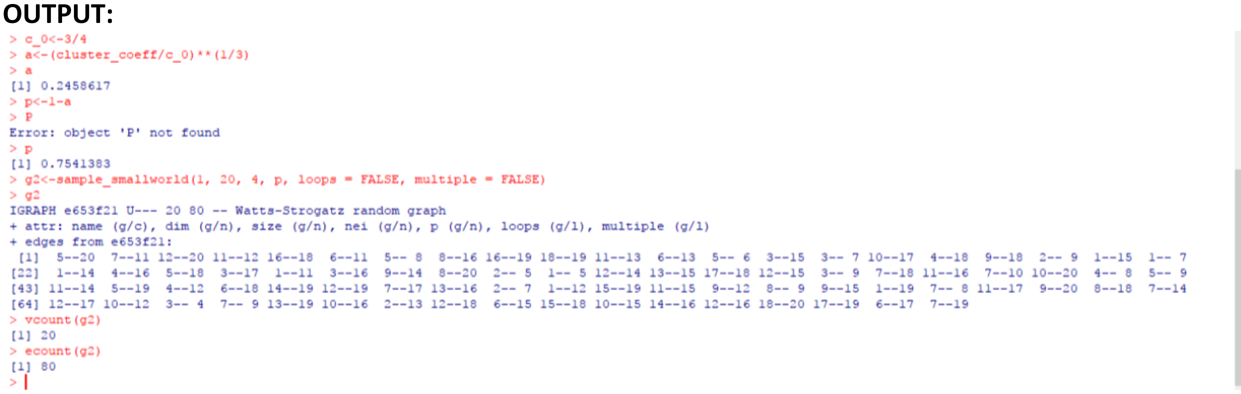
OUTPUT:


```
> g1<- erdos.renyi.game(20, p, type = c("gnp", "gnm"), directed = FALSE,loops=FALSE)
> g1
IGRAPH 24f0967 U--- 20 143 -- Erdos renyi (gnp) graph
+ attr: name (g/c), type (g/c), loops (g/l), p (g/n)
+ edges from 24f0967:
  [1]  1-- 2  1-- 3  2-- 3  1-- 4  3-- 4  1-- 5  2-- 5  3-- 5  1-- 6  2-- 6  3-- 6  4-- 6  1-- 7  2-- 7  3-- 7  6-- 7  2-- 8  4-- 8  5-- 8  6-- 8  7-- 8
 [22]  1-- 9  2-- 9  3-- 9  4-- 9  5-- 9  7-- 9  8-- 9  2--10  3--10  6--10  7--10  9--10  1--11  3--11  4--11  5--11  9--11 10--11  1--12  3--12  5--12
 [43]  6--12  7--12  9--12 10--12 11--12  1--13  2--13  3--13  4--13  5--13  6--13  7--13  8--13  9--13 10--13 11--13 12--13  1--14  3--14  4--14  5--14
 [64]  6--14  7--14  8--14  9--14 10--14 11--14 12--14 13--14  2--15  3--15  4--15  6--15  8--15  9--15 10--15 11--15 12--15 13--15 14--15  2--16  3--16
 [85]  4--16  5--16  7--16  8--16 10--16 11--16 12--16 13--16 15--16  1--17  2--17  3--17  4--17  5--17  6--17  7--17  8--17 10--17 12--17 13--17 15--17
[106] 16--17  1--18  2--18  3--18  4--18  5--18  6--18 13--18 14--18 15--18 16--18 17--18  2--19  3--19  6--19  7--19  8--19  9--19 12--19 13--19 14--19
[127] 15--19 16--19 17--19 18--19  1--20  2--20  3--20  4--20  5--20  9--20 10--20 11--20 12--20 13--20 15--20 17--20 19--20
> vcount(g1)
[1] 20
> ecount(g1)
[1] 143
> |
```

5) Simulate the given social network using small world properties model for 20 nodes. Compute the beta value from C(p) = (1-p)^3 * C(0), in which beta = p, C(p)= clustering coefficent of given network, C(0) is clustering coefficent of regular lattice .
CODE:
> vcount(g1)
[1] 20
> ecount(g1)

```
[1] 0
> c_0<-3/4
> a<-(cluster_coeff/c_0)**(1/3)
> a
[1] 0.2458617
> p<-1-a
> p
[1] 0.7541383
> g2<-sample_smallworld(1, 20, 4, p, loops = FALSE, multiple = FALSE)
> g2
IGRAPH e653f21 U--- 20 80 -- Watts-Strogatz random graph
+ attr: name (g/c), dim (g/n), size (g/n), nei (g/n), p (g/n), loops (g/l), multiple (g/l)
+ edges from e653f21:
 [1]  5--20  7--11 12--20 11--12 16--18  6--11  5-- 8  8--16 16--19 18--19 11--13  6--13  5-- 6  3--15  3-- 7 10--17  4--18  9--18  2-- 9  1--15  1-- 7
[22]  1--14  4--16  5--18  3--17  1--11  3--16  9--14  8--20  2-- 5  1-- 5 12--14 13--15 17--18 12--15  3-- 9  7--18 11--16  7--10 10--20  4-- 8  5-- 9
[43] 11--14  5--19  4--12  6--18 14--19 12--19  7--17 13--16  2-- 7  1--12 15--19 11--15  9--12  8-- 9  9--15  1--19  7-- 8 11--17  9--20  8--18  7--14
[64] 12--17 10--12  3-- 4  7-- 9 13--19 10--16  2--13 12--18  6--15 15--18 10--15 14--16 12--16 18--20 17--19  6--17  7--19
> vcount(g2)
[1] 20
> ecount(g2)
[1] 80
>
```

OUTPUT:



```
> c_0<-3/4
> a<-(cluster_coeff/c_0)**(1/3)
> a
[1] 0.2458617
> p<-1-a
> P
Error: object 'P' not found
> p
[1] 0.7541383
> g2<-sample_smallworld(1, 20, 4, p, loops = FALSE, multiple = FALSE)
> g2
IGRAPH e653f21 U--- 20 80 -- Watts-Strogatz random graph
+ attr: name (g/c), dim (g/n), size (g/n), nei (g/n), p (g/n), loops (g/l), multiple (g/l)
+ edges from e653f21:
 [1]  5--20  7--11 12--20 11--12 16--18  6--11  5-- 8  8--16 16--19 18--19 11--13  6--13  5-- 6  3--15  3-- 7 10--17  4--18  9--18  2-- 9  1--15  1-- 7
[22]  1--14  4--16  5--18  3--17  1--11  3--16  9--14  8--20  2-- 5  1-- 5 12--14 13--15 17--18 12--15  3-- 9  7--18 11--16  7--10 10--20  4-- 8  5-- 9
[43] 11--14  5--19  4--12  6--18 14--19 12--19  7--17 13--16  2-- 7  1--12 15--19 11--15  9--12  8-- 9  9--15  1--19  7-- 8 11--17  9--20  8--18  7--14
[64] 12--17 10--12  3-- 4  7-- 9 13--19 10--16  2--13 12--18  6--15 15--18 10--15 14--16 12--16 18--20 17--19  6--17  7--19
> vcount(g2)
[1] 20
> ecount(g2)
[1] 80
>
```

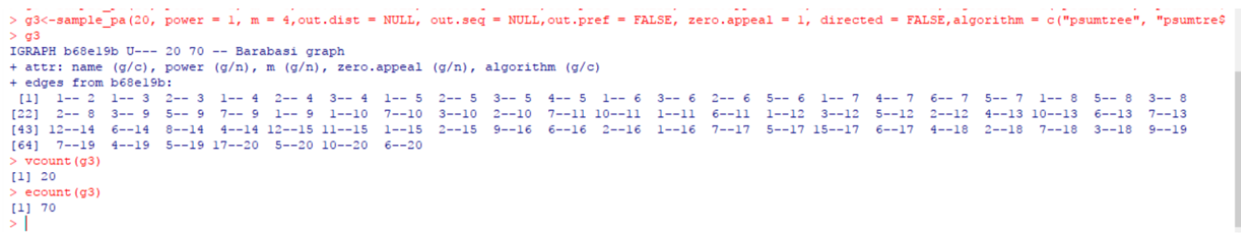
6) Simulate the given social network using preferential attachment model for 20 nodes with expected degree m computed from real networks.

CODE:

```
> g3<-sample_pa(20, power = 1, m = 4,out.dist = NULL, out.seq = NULL,out.pref = FALSE, zero.appeal = 1, directed = FALSE,algorithm = c("psumtree", "psumtree-multiple", "bag"), start.graph = NULL)
> g3
IGRAPH b68e19b U--- 20 70 -- Barabasi graph
+ attr: name (g/c), power (g/n), m (g/n), zero.appeal (g/n), algorithm (g/c)
```

```
+ edges from b68e19b:
 [1]  1-- 2  1-- 3  2-- 3  1-- 4  2-- 4  3-- 4  1-- 5  2-- 5  3-- 5  4-- 5  1-- 6  3-- 6  2-- 6  5-- 6  1-- 7  4-- 7  6-- 7  5--
7  1-- 8  5-- 8  3-- 8
[22]  2-- 8  3-- 9  5-- 9  7-- 9  1-- 9  1--10  7--10  3--10  2--10  7--11 10--11  1--11  6--11  1--12  3--12  5--12
2--12  4--13 10--13  6--13  7--13
[43] 12--14  6--14  8--14  4--14 12--15 11--15  1--15  2--15  9--16  6--16  2--16  1--16  7--17  5--17 15--17
6--17  4--18  2--18  7--18  3--18  9--19
[64]  7--19  4--19  5--19 17--20  5--20 10--20  6--20
> vcount(g3)
[1] 20
> ecount(g3)
[1] 70
>
```

OUTPUT:



```
> g3<-sample_pa(20, power = 1, m = 4,out.dist = NULL, out.seq = NULL,out.pref = FALSE, zero.appeal = 1, directed = FALSE,algorithm = c("psumtree", "psumtre$
> g3
IGRAPH b68e19b U--- 20 70 -- Barabasi graph
+ attr: name (g/c), power (g/n), m (g/n), zero.appeal (g/n), algorithm (g/c)
+ edges from b68e19b:
 [1]  1-- 2  1-- 3  2-- 3  1-- 4  2-- 4  3-- 4  1-- 5  2-- 5  3-- 5  4-- 5  1-- 6  3-- 6  2-- 6  5-- 6  1-- 7  4-- 7  6-- 7  5-- 7  1-- 8  5-- 8  3-- 8
[22]  2-- 8  3-- 9  5-- 9  7-- 9  1-- 9  1--10  7--10  3--10  2--10  7--11 10--11  1--11  6--11  1--12  3--12  5--12  2--12  4--13 10--13  6--13  7--13
[43] 12--14  6--14  8--14  4--14 12--15 11--15  1--15  2--15  9--16  6--16  2--16  1--16  7--17  5--17 15--17  6--17  4--18  2--18  7--18  3--18  9--19
[64]  7--19  4--19  5--19 17--20  5--20 10--20  6--20
> vcount(g3)
[1] 20
> ecount(g3)
[1] 70
>
```

7) Compare the clustering coefficent and average path length properties of given real network and simulated networks.

CODE:

```
> cluster_coeff_real<-transitivity(g, type = c("undirected", "global", "globalundirected",
"localundirected", "local", "average", "localaverage","localaverageundirected", "barrat", "weighted"),
vids = NULL,weights = NULL, isolates = c("NaN", "zero"))
> cluster_coeff_real
[1] 0.01114638
> cluster_coeff_g1<-transitivity(g1, type = c("undirected", "global", "globalundirected",
"localundirected", "local", "average", "localaverage","localaverageundirected", "barrat", "weighted"),
vids = NULL,weights = NULL, isolates = c("NaN", "zero"))
> cluster_coeff_g1
[1] 0.7440883
> cluster_coeff_g2<-transitivity(g2, type = c("undirected", "global", "globalundirected",
"localundirected", "local", "average", "localaverage","localaverageundirected", "barrat", "weighted"),
vids = NULL,weights = NULL, isolates = c("NaN", "zero"))
> cluster_coeff_g2
[1] 0.4159869
> cluster_coeff_g3<-transitivity(g3, type = c("undirected", "global", "globalundirected",
"localundirected", "local", "average", "localaverage","localaverageundirected", "barrat", "weighted"),
vids = NULL,weights = NULL, isolates = c("NaN", "zero"))
> cluster_coeff_g3
[1] 0.4235977
>
```

> average.path.length(g)
[1] 3.842426
> average.path.length(g1)
[1] 1.252632
> average.path.length(g2)
[1] 1.589474
> average.path.length(g3)
[1] 1.668421
>

OUTPUT:

```
> cluster_coeff_real<-transitivity(g, type = c("undirected", "global", "globalundirected", "localundirected", "local", "average", "localaverage","localaver$
> cluster_coeff_real
[1] 0.01114638
> cluster_coeff_g1<-transitivity(g1, type = c("undirected", "global", "globalundirected", "localundirected", "local", "average", "localaverage","localavera$
> cluster_coeff_g1
[1] 0.7440883
> cluster_coeff_g2<-transitivity(g2, type = c("undirected", "global", "globalundirected", "localundirected", "local", "average", "localaverage","localavera$
> cluster_coeff_g2
[1] 0.4159869
> cluster_coeff_g3<-transitivity(g3, type = c("undirected", "global", "globalundirected", "localundirected", "local", "average", "localaverage","localavera$
> cluster_coeff_g3
[1] 0.4235977
> average.path.length(g)
[1] 3.842426
> average.path.length(g1)
[1] 1.252632
> average.path.length(g2)
[1] 1.589474
> average.path.length(g3)
[1] 1.668421
> |
```

8) Show the average degree distribution for preferential attachment model follows power law distribution.

CODE:

> g3_avg_deg_dist<-mean(degree_distribution(g3, cumulative = FALSE))
> g3_avg_deg_dist
[1] 0.07142857
> all.deg.testgraph<-degree(g3,v=V(g3),mode="all")
> all.deg.testgraph
 [1] 13 11 10 9 12 11 11 5 6 7 5 6 4 4 5 4 5 4 4 4
> power<-power.law.fit(all.deg.testgraph)
> power
$continuous
[1] FALSE

$alpha
[1] 10.00893

$xmin
[1] 11

$logLik
[1] -5.469348

$KS.stat

[1] 0.05753324

$KS.p
[1] 1

>

OUTPUT:

```
> g3_avg_deg_dist<-mean(degree_distribution(g3, cumulative = FALSE))
> g3_avg_deg_dist
[1] 0.07142857
> all.deg.testgraph<-degree(g3,v=V(g3),mode="all")
> all.deg.testgraph
 [1] 13 11 10  9 12 11 11  5  6  7  5  6  4  4  5  4  5  4  4  4
> power<-power.law.fit(all.deg.testgraph)
> power
$continuous
[1] FALSE

$alpha
[1] 10.00893

$xmin
[1] 11

$logLik
[1] -5.469348

$KS.stat
[1] 0.05753324

$KS.p
[1] 1
```

9) With the simulated preferential attachment model graph in 6, detect the communities using any one of the node similarity based method and edge betweenness method.

CODE:

> similarity(g3, vids = V(g3), mode = c("all", "out", "in", "total"),loops = FALSE, method = c("jaccard", "dice", "invlogweighted"))

```
          [,1]      [,2]      [,3]       [,4]      [,5]       [,6]       [,7]      [,8]      [,9]     [,10]     [,11]     [,12]      [,13]     [,14]
 [1,] 1.0000000 0.6000000 0.5333333 0.22222222 0.4705882 0.33333333 0.33333333 0.2000000
0.2666667 0.2500000 0.2857143 0.2666667 0.30769231 0.3076923
 [2,] 0.6000000 1.0000000 0.6153846 0.25000000 0.3529412 0.22222222 0.37500000 0.2307692
0.3076923 0.1250000 0.3333333 0.3076923 0.25000000 0.3636364
 [3,] 0.5333333 0.6153846 1.0000000 0.26666667 0.4666667 0.16666667 0.50000000 0.2500000
0.1428571 0.1333333 0.2500000 0.2307692 0.27272727 0.4000000
 [4,] 0.2222222 0.2500000 0.2666667 1.00000000 0.3125000 0.53846154 0.33333333 0.5555556
0.5000000 0.4545455 0.1666667 0.5000000 0.08333333 0.0000000
 [5,] 0.4705882 0.3529412 0.4666667 0.31250000 1.0000000 0.35294118 0.35294118 0.2142857
0.2857143 0.3571429 0.2142857 0.2000000 0.23076923 0.3333333
 [6,] 0.3333333 0.2222222 0.1666667 0.53846154 0.3529412 1.00000000 0.29411765 0.4545455
0.4166667 0.6363636 0.1428571 0.4166667 0.07142857 0.0000000
 [7,] 0.3333333 0.3750000 0.5000000 0.33333333 0.3529412 0.29411765 1.00000000 0.1428571
0.2142857 0.2000000 0.2307692 0.1333333 0.25000000 0.1538462
 [8,] 0.2000000 0.2307692 0.2500000 0.55555556 0.2142857 0.45454545 0.14285714 1.0000000
0.3750000 0.3333333 0.1111111 0.8333333 0.00000000 0.0000000
 [9,] 0.2666667 0.3076923 0.1428571 0.50000000 0.2857143 0.41666667 0.21428571 0.3750000
1.0000000 0.3000000 0.2222222 0.3333333 0.11111111 0.0000000
```

```
[10,] 0.2500000 0.1250000 0.1333333 0.45454545 0.3571429 0.63636364 0.20000000 0.3333333
0.3000000 1.0000000 0.2000000 0.3000000 0.10000000 0.0000000
[11,] 0.2857143 0.3333333 0.2500000 0.16666667 0.2142857 0.14285714 0.23076923 0.1111111
0.2222222 0.2000000 1.0000000 0.2222222 0.50000000 0.1250000
[12,] 0.2666667 0.3076923 0.2307692 0.50000000 0.2000000 0.41666667 0.13333333 0.8333333
0.3333333 0.3000000 0.2222222 1.0000000 0.00000000 0.0000000
[13,] 0.3076923 0.2500000 0.2727273 0.08333333 0.2307692 0.07142857 0.25000000 0.0000000
0.1111111 0.1000000 0.5000000 0.0000000 1.00000000 0.3333333
[14,] 0.3076923 0.3636364 0.4000000 0.00000000 0.3333333 0.00000000 0.15384615 0.0000000
0.0000000 0.0000000 0.1250000 0.0000000 0.33333333 1.0000000
[15,] 0.2000000 0.1428571 0.2500000 0.16666667 0.3076923 0.33333333 0.23076923 0.2500000
0.1000000 0.3333333 0.1111111 0.2222222 0.00000000 0.1250000
[16,] 0.2142857 0.1538462 0.4000000 0.18181818 0.3333333 0.15384615 0.25000000 0.2857143
0.1111111 0.2222222 0.2857143 0.2500000 0.14285714 0.1428571
[17,] 0.2857143 0.2307692 0.1538462 0.16666667 0.2142857 0.23076923 0.14285714 0.1111111
0.2222222 0.2000000 0.4285714 0.2222222 0.28571429 0.1250000
[18,] 0.3076923 0.1538462 0.1666667 0.30000000 0.3333333 0.25000000 0.07142857 0.2857143
0.2500000 0.3750000 0.1250000 0.2500000 0.33333333 0.1428571
[19,] 0.3076923 0.1538462 0.2727273 0.18181818 0.2307692 0.15384615 0.25000000 0.1250000
0.2500000 0.1000000 0.1250000 0.1111111 0.33333333 0.1428571
[20,] 0.2142857 0.2500000 0.2727273 0.08333333 0.1428571 0.15384615 0.36363636 0.1250000
0.1111111 0.0000000 0.2857143 0.1111111 0.33333333 0.1428571
          [,15]     [,16]     [,17]      [,18]     [,19]      [,20]
 [1,] 0.2000000 0.2142857 0.2857143 0.30769231 0.3076923 0.21428571
 [2,] 0.1428571 0.1538462 0.2307692 0.15384615 0.1538462 0.25000000
 [3,] 0.2500000 0.4000000 0.1538462 0.16666667 0.2727273 0.27272727
 [4,] 0.1666667 0.1818182 0.1666667 0.30000000 0.1818182 0.08333333
 [5,] 0.3076923 0.3333333 0.2142857 0.33333333 0.2307692 0.14285714
 [6,] 0.3333333 0.1538462 0.2307692 0.25000000 0.1538462 0.15384615
 [7,] 0.2307692 0.2500000 0.1428571 0.07142857 0.2500000 0.36363636
 [8,] 0.2500000 0.2857143 0.1111111 0.28571429 0.1250000 0.12500000
 [9,] 0.1000000 0.1111111 0.2222222 0.25000000 0.2500000 0.11111111
[10,] 0.3333333 0.2222222 0.2000000 0.37500000 0.1000000 0.00000000
[11,] 0.1111111 0.2857143 0.4285714 0.12500000 0.1250000 0.28571429
[12,] 0.2222222 0.2500000 0.2222222 0.25000000 0.1111111 0.11111111
[13,] 0.0000000 0.1428571 0.2857143 0.33333333 0.3333333 0.33333333
[14,] 0.1250000 0.1428571 0.1250000 0.14285714 0.1428571 0.14285714
[15,] 1.0000000 0.2857143 0.0000000 0.12500000 0.0000000 0.12500000
[16,] 0.2857143 1.0000000 0.1250000 0.14285714 0.1428571 0.14285714
[17,] 0.0000000 0.1250000 1.0000000 0.12500000 0.2857143 0.28571429
[18,] 0.1250000 0.1428571 0.1250000 1.00000000 0.3333333 0.00000000
[19,] 0.0000000 0.1428571 0.2857143 0.33333333 1.0000000 0.14285714
[20,] 0.1250000 0.1428571 0.2857143 0.00000000 0.1428571 1.00000000
> edge_betweenness(g3, e = E(g3), directed = TRUE, weights = NULL)
 [1] 1.916667 2.309524 1.833333 4.877579 4.365079 3.253770 2.966667 3.724242 2.850433 4.362933
3.553770 4.704203 4.298846 3.605357 4.292857 3.470437
[17] 3.920869 3.900433 5.401389 5.945833 3.334722 3.938889 4.126190 4.316667 4.834524 4.750000
4.398413 4.471068 4.465512 4.989322 5.516667 3.031746
```

[33] 4.916667 5.115079 4.401389 3.668056 5.795833 3.438889 4.981746 3.853968 6.424603 4.933333 4.087500 8.377579 3.087500 6.266468 3.683333 3.350000
[49] 5.983333 5.700000 3.345833 6.595833 5.250000 4.833333 5.940909 5.433333 4.616667 5.250000 3.590909 5.606061 6.339827 4.082251 2.945833 5.550000
[65] 4.545833 6.483333 2.840909 7.813636 4.421140 5.721140
>

OUTPUT:

```
> similarity(g3, vids = V(g3), mode = c("all", "out", "in", "total"),loops = FALSE, method = c("jaccard", "dice", "invlogweighted"))
           [,1]      [,2]      [,3]       [,4]      [,5]       [,6]       [,7]      [,8]      [,9]     [,10]     [,11]     [,12]      [,13]     [,14]
 [1,] 1.0000000 0.6000000 0.5333333 0.22222222 0.4705882 0.33333333 0.33333333 0.2000000 0.2666667 0.2500000 0.2857143 0.2666667 0.30769231 0.3076923
 [2,] 0.6000000 1.0000000 0.6153846 0.25000000 0.3529412 0.22222222 0.37500000 0.2307692 0.3076923 0.1250000 0.3333333 0.3076923 0.25000000 0.3636364
 [3,] 0.5333333 0.6153846 1.0000000 0.26666667 0.4666667 0.16666667 0.50000000 0.2500000 0.1428571 0.1333333 0.2500000 0.2307692 0.27272727 0.4000000
 [4,] 0.2222222 0.2500000 0.2666667 1.00000000 0.3125000 0.53846154 0.33333333 0.5555556 0.5000000 0.4545455 0.1666667 0.5000000 0.08333333 0.0000000
 [5,] 0.4705882 0.3529412 0.4666667 0.31250000 1.0000000 0.35294118 0.35294118 0.2142857 0.2857143 0.3571429 0.2142857 0.2000000 0.23076923 0.3333333
 [6,] 0.3333333 0.2222222 0.1666667 0.53846154 0.3529412 1.00000000 0.29411765 0.4545455 0.4166667 0.6363636 0.1428571 0.4166667 0.07142857 0.0000000
 [7,] 0.3333333 0.3750000 0.5000000 0.33333333 0.3529412 0.29411765 1.00000000 0.1428571 0.2142857 0.2000000 0.2307692 0.1333333 0.25000000 0.1538462
 [8,] 0.2000000 0.2307692 0.2500000 0.55555556 0.2142857 0.45454545 0.14285714 1.0000000 0.3750000 0.3333333 0.1111111 0.8333333 0.00000000 0.0000000
 [9,] 0.2666667 0.3076923 0.1428571 0.50000000 0.2857143 0.41666667 0.21428571 0.3750000 1.0000000 0.3000000 0.2222222 0.3333333 0.11111111 0.0000000
[10,] 0.2500000 0.1250000 0.1333333 0.45454545 0.3571429 0.63636364 0.20000000 0.3333333 0.3000000 1.0000000 0.2000000 0.3000000 0.10000000 0.0000000
[11,] 0.2857143 0.3333333 0.2500000 0.16666667 0.2142857 0.14285714 0.23076923 0.1111111 0.2222222 0.2000000 1.0000000 0.2222222 0.50000000 0.1250000
[12,] 0.2666667 0.3076923 0.2307692 0.50000000 0.2000000 0.41666667 0.13333333 0.8333333 0.3333333 0.3000000 0.2222222 1.0000000 0.00000000 0.0000000
[13,] 0.3076923 0.2500000 0.2727273 0.08333333 0.2307692 0.07142857 0.25000000 0.0000000 0.1111111 0.1000000 0.5000000 0.0000000 1.00000000 0.3333333
[14,] 0.3076923 0.3636364 0.4000000 0.00000000 0.3333333 0.00000000 0.15384615 0.0000000 0.0000000 0.0000000 0.1250000 0.0000000 0.33333333 1.0000000
[15,] 0.2000000 0.1428571 0.2500000 0.16666667 0.3076923 0.33333333 0.23076923 0.2500000 0.1000000 0.3333333 0.1111111 0.2222222 0.00000000 0.1250000
[16,] 0.2142857 0.1538462 0.4000000 0.18181818 0.3333333 0.15384615 0.25000000 0.2857143 0.1111111 0.2222222 0.2857143 0.2500000 0.14285714 0.1428571
[17,] 0.2857143 0.2307692 0.1538462 0.16666667 0.2142857 0.23076923 0.14285714 0.1111111 0.2222222 0.2000000 0.4285714 0.2222222 0.28571429 0.1250000
[18,] 0.3076923 0.1538462 0.1666667 0.30000000 0.3333333 0.25000000 0.07142857 0.2857143 0.2500000 0.3750000 0.1250000 0.2500000 0.33333333 0.1428571
[19,] 0.3076923 0.1538462 0.2727273 0.18181818 0.2307692 0.15384615 0.25000000 0.1250000 0.2500000 0.1000000 0.1250000 0.1111111 0.33333333 0.1428571
[20,] 0.2142857 0.2500000 0.2727273 0.08333333 0.1428571 0.15384615 0.36363636 0.1250000 0.1111111 0.0000000 0.2857143 0.1111111 0.33333333 0.1428571
          [,15]     [,16]     [,17]      [,18]     [,19]      [,20]
 [1,] 0.2000000 0.2142857 0.2857143 0.30769231 0.3076923 0.21428571
 [2,] 0.1428571 0.1538462 0.2307692 0.15384615 0.1538462 0.25000000
 [3,] 0.2500000 0.4000000 0.1538462 0.16666667 0.2727273 0.27272727
 [4,] 0.1666667 0.1818182 0.1666667 0.30000000 0.1818182 0.08333333
 [5,] 0.3076923 0.3333333 0.2142857 0.33333333 0.2307692 0.14285714
 [6,] 0.3333333 0.1538462 0.2307692 0.25000000 0.1538462 0.15384615
 [7,] 0.2307692 0.2500000 0.1428571 0.07142857 0.2500000 0.36363636
 [8,] 0.2500000 0.2857143 0.1111111 0.28571429 0.1250000 0.12500000
 [9,] 0.1000000 0.1111111 0.2222222 0.25000000 0.2500000 0.11111111
[10,] 0.3333333 0.2222222 0.2000000 0.37500000 0.1000000 0.00000000
[11,] 0.1111111 0.2857143 0.4285714 0.12500000 0.1250000 0.28571429
[12,] 0.2222222 0.2500000 0.2222222 0.25000000 0.1111111 0.11111111
[13,] 0.0000000 0.1428571 0.2857143 0.33333333 0.3333333 0.33333333
[14,] 0.1250000 0.1428571 0.1250000 0.14285714 0.1428571 0.14285714
[15,] 1.0000000 0.2857143 0.0000000 0.12500000 0.0000000 0.12500000
[16,] 0.2857143 1.0000000 0.1250000 0.14285714 0.1428571 0.14285714
[17,] 0.0000000 0.1250000 1.0000000 0.12500000 0.2857143 0.28571429
[18,] 0.1250000 0.1428571 0.1250000 1.00000000 0.3333333 0.00000000
[19,] 0.0000000 0.1428571 0.2857143 0.33333333 1.0000000 0.14285714
[20,] 0.1250000 0.1428571 0.2857143 0.00000000 0.1428571 1.00000000
> edge_betweenness(g3, e = E(g3), directed = TRUE, weights = NULL)
 [1] 1.916667 2.309524 1.833333 4.877579 4.365079 3.253770 2.966667 3.724242 2.850433 4.362933 3.553770 4.704203 4.298846 3.605357 4.292857 3.470437
[17] 3.920869 3.900433 5.401389 5.945833 3.334722 3.938889 4.126190 4.316667 4.834524 4.750000 4.398413 4.471068 4.465512 4.989322 5.516667 3.031746
[33] 4.916667 5.115079 4.401389 3.668056 5.795833 3.438889 4.981746 3.853968 6.424603 4.933333 4.087500 8.377579 3.087500 6.266468 3.683333 3.350000
[49] 5.983333 5.700000 3.345833 6.595833 5.250000 4.833333 5.940909 5.433333 4.616667 5.250000 3.590909 5.606061 6.339827 4.082251 2.945833 5.550000
[65] 4.545833 6.483333 2.840909 7.813636 4.421140 5.721140
> 
```