In this homework, we will gain experience with CUG testing, QAP testing, and exponential random graph modeling using the familiar Lord of the Rings 3 network.

You will need the LOTR3 network, the invlogit function from the lab, and the mycugtest and myqaptest utilities. These have been packaged in `hwk6.zip`.

Part I: Attribute setup

Create a new attribute that simplifies the nine character Types in the network into three categories using the following table:

```
typeTable <- data.frame(Type=c("Dwarf", "Elf", "Ent", "Hobbit", "Human",
"Orc", "Undead", "Unknown", "Wizard"),
     Category=c("Fellow", "Fellow", "Other", "Fellow", "Fellow", "Evil",
"Evil", "Other", "Fellow"))
```

Create a data frame of the Type attributes from the network. Then use the dplyr inner_join function to compute the correct category for each character. Create the attribute Category for each node using the result of the join. Important: You must use as.character when setting the attribute values because the data frame contains factors and igraph attributes cannot contain factors.

For you Tolkien fans out there, yes, I am aware that Saruman and Sauron are both wizards and not part of the fellowship, but they don't appear in this network, so we are OK.

Part II: CUG and QAP

For each CUG test, display the summary (`print.cug.test`) and the probability plot (`plot.cug.test`), state the null hypothesis, and indicate whether the hypothesis is accepted or rejected.

- assortativity of the network by the original Type attribute
  - remember to convert the attributes to a factor when calling the function.
- assortativity of the network by the created Category attribute

For each QAP test, display the summary (`print.qaptest`) and the probability plot (`plot.qaptest`), state the null hypothesis, and indicate whether the hypothesis is accepted or rejected.

- assortativity of the network by Type
- assortativity of the network by Category

Part III: ERGM

For each model, display the model summary, the MCMC diagnostics (where applicable), and the network goodness-of-fit with plots. For the MCMC models (all except #1), save the model fit in an Rdata file and load it when running knitr, include the call to ergm as a non-executing code block as we did in lab.

Create four models of the LOTR3 network.

- Model 1: edges and Category assortativity
  - note that there are three values of this attribute (yielding six possible parameters) but there are no examples of Evil-Other or Other-Other edges, so these must be included in the "base": `base=c(4,6)`
  - note that the fit to the degree distribution is poor, especially for degree=1
  - note also that the non-matching version of the nodemix parameters were not very strong, so matching may be enough
- Model 2: edges, Category match (nodematch) and degree(1)
  - note that the esp distribution is a poor match
- Model 3: Model 2 + gwesp(cutoff=7)
  - the cutoff parameter controls how many esp terms are calculated
  - note that the degree term now has poor fit -- it seems to be redundant with the gwesp term
  - note also that degree 4 is a particularly big mismatch
- Model 4: edges, Category nodemix but with only the matching values for Evil-Evil and Fellow-Fellow base=c(2,4,5,6), gwesp(cutoff=7), and degree(2:4)
  - coefficients are significant, except for degree 2 and 3. We are keeping these because they improve the fit of the degree distribution
  - note that there is inadequate mixing time in this computation (autocorrelation is high)
- Model 5: Model 4 increasing burnin to 100000 and increasing interval to 5000 and samplesize to 2048. (These were the final values from lab.)

Part IV: ERGM Interpretation

- Compare the probability of an Evil-Evil edge versus a Fellow-Fellow edge vs an edge not of these types.
  - Fill in the following data frame:
  -

Case	edges	nodemix-EE	nodemix-FF	Log-odds	Cond. probability
Evil-Evil edge					

Fellow-Fellow edge
Other edge

- Question: What does the model fit show about assortativity by category in this network? How does this add to what we learned from CUG and QAP testing?

- Compute the probability of a second edge being added to a evil character to another evil character considering all the possible degree values within the model coefficients: six cases.
  - Create a data frame with the six cases, their log-odds and probabilities
  - Create a side-by-side barplot comparing the conditional probabilities across conditions.
  - Question: What does the model fit show about degree variation