Exercise 2

This exercise investigates a multidimensional CAT with **mirtCAT** using off-line components from a previously estimated model. As well, Monte Carlo simulation aspects are constructed to demonstrate the general behavior of the multidimensional CAT.

Question 1

The **mirt** model is available in the `Exercise_02.Rdata` workspace file. Read this file into R with `load('Exercise_02.Rdata')`, and inspect the object workspace.

Given that all the hard estimation work has been done, all you get to do if have fun with **mirtCAT**! Familiarize yourself with the input objects to get a better understanding of what you're working by using functions such as `coef()`, `plot()`, etc.

Question 2

Generate 8 independent response patterns according to the population ability sets θ = [-1,-1]; [-1, 0]; [-1,1]; [0,-1]; [0,0]; [0,1]; [1,-1]; and [1,1] using the `generate_pattern()` function.

Question 3

Given the previously generated patterns, pass this object to `mirtCAT()` to run the multidimensional CAT sessions. Use the selection criteria `DPrule` throughout (including the initial item), and terminate the test when both ability estimates have $SE(\hat{\theta}) < .35$.

When completed, plot the results of each response pattern. Finally, create a simple `table()` indicating how often the items were selected in the response patterns (hint: use the `summary()` function on each element in the `mirtCAT()` results, and extract the suitable element).

Question 4

The previous question's simulations were a bit on the slow side, and obviously will grow if more response patterns are studied or more intensive CAT sessions are performed. To circumvent some of these issues, try using the build-in parallelization features for Monte Carlo simulations.

Load the `parallel` package and define a cluster object using `cl <- makeCluster(2)`. Note that Windows users will likely need to allow permission for OS parallelization if this is the first time you've used parallel processing in R. Next, supply the `cl` object to the syntax used in Question 3, and rerun the simulation.

When done, remove the cluster object with `stopCluster(cl)`.