Everything is contained in the attached zip. The following is a brief rundown.

maxSharpe.R and maxSharpeRobust.R are the same scripts as last project.

maxSharpeConstrained.R is a script that can process asset constraints, sector constraints, and class constraints. You can call it as follows

Rscript --vanilla maxSharpeConstrained.R ./returns.csv ./assetConstraints.csv ./sectorConstraints.csv ./classConstraints.csv

where the csv's follow the same format as the files contained in the testData folder. It will output a csv with weights and another with performance metrics, exactly the same as maxSharpe.R. You can leave constraints blank if you don't want them to be active. The less constraints the faster the program runs.

maxSharpeConstrainedLoop.R runs maxSharpeConstrained.R many times on a set of strategies you define. Specifically, you must fill a folder the same way as the folder loopTestData; containing one returns.csv but many different constraint files according to your strategy. You must name them the way I did in my folder (i.e. assetConstraints_1.csv, sectorConstraints_2.csv, and so forth for as many strategies as you'll be considering). Then, you call the script as follows

Rscript --vanilla maxSharpeConstrainedLoop.R ./loopTestData numberOfStrategies

In the example I provided, numberOfStrategies is 2. It will then run the algorithm through your strategies and output a file like ./sharpeLoops.csv with, as you requested, has strategies along the columns and metrics down the rows.

In all these scripts the algorithm takes returns and all constraints as raw numbers (so that if you multiplied them by 100, you get percent) and also outputs the weights as raw numbers. This way the allocation size doesn't affect how the constraints are interpreted: A minimum of .3 or 30% on asset 1 is the same irrespective of whether you're investing $10k or $1M. You can then multiply the weights by the investment amount and get the per-asset dollar allocations back. This approach is more general than specifying a different sum for every application and is typically the way it's done in the investment world. If you'd rather specify a sum for each portfolio, let me know and I can adjust the scripts accordingly.

A final note: The assets/securities must have text names (column headers) or else R has some issues. That's why I changed all the names in my test data to be "X1001" and so on.