Tasks to be completed for project:

April 25 (Sat): Uploaded bare-bones code to repo; generalised from Bayesian Inference assignment to have multiple parameters and arbitrary posteriors/priors - Kyle

May 4 (Mon): presentation review (all) - to be on the earlier (safer) side

May 5 and 6 (Tue-Wed): buffer days

May 7 (Thu): in-class presentations (all)

May 8 (Fri): delivery of code and presentation slides

Use basic algorithm to run different tests over and over; all “physics functions” and priors, etc as separate scripts with functions that can be imported

Steps for Task 1:

1. Have basic posterior sampling, chain, and convergence test running to use to test everything else **- Kyle; aim for end of Sun April 26**
2. Globally import data table from file **- Pratik**
3. Def mu_model (for 3 different cases of OmegaK) **- Pratik/Sudheer; try to have it done after OH on Monday, April 27**
    a. OmegaK is based on OmegaM and OmegaLambda
    b. Analytically check where degeneracy appears (discuss during office hours)?
4. Define a function for the priors on all 4 parameters (OmegaM, OmegaLambda, H0, and M) such that M can have a uniform or a gaussian prior **- Avinash;**
   try to have it done by Sun April 26
    a. Uniform for all
5. Define likelihood functions **- Avinash; try to have it done by Sun April 26**
    a. 1likelihood function - with and without systematic errors (systematic errors: using covariance matrix from additional data file; no systematic error: just diagonal terms using error from the main dataset)
6. Define posterior function for the chain to sample from **- Kyle**
    a. Based on likelihood and priors
7. Create and propagate chain using MH and sampling from posteriors **- Kyle**
    a. Gaussian generating function - independent for each parameter
8. Define a convergence test for the MCMC **- Kyle**
9. Marginalise (integrate numerically) over H0 and M to get posterior only in terms of OmegaM and OmegaLambda **- Sudheer**
10. Plot posterior for OmegaM and OmegaLambda **- Sudheer; aim for Friday May 1**
    a. Include both analyses (with and without errors) in the same plot

Steps for Task 2:

11. Repeat same analysis as Task 1 except now marginalise over OmegaM, OmegaLambda, and M **- Pratik**

---

def mu_model(param, data)
    params = array of OmegaM, OmegaLambda, H0, M
    data = array of m, z, merr for each SN
    returns mu_model for each SN (array of length 40)