```
# SE 201B: NONLINEAR STRUCTURAL ANALYSIS (WI 2021)
# HOMEWORK # 1
# NONLINEAR QUASI-STATIC & TIME-HISTORY ANALYSIS OF A SDOF SYSTEM
# ####################################################################################
# Angshuman Deb

# TIME-HISTORY ANALYSIS -------------------------------------------------------------

# SET UP GROUND-MOTION-ANALYSIS PARAMETERS --------------------------------------------
set gmDirection 1;                                          # ground-motion direction (Need to set it manually)
set gmFact 1.0;                                             # ground-motion scaling factor (Need to set it manually)
set gmFileName "SYL360.txt";                                # ground motion file name with two cols (time and u_g_ddot) (Need to set it manually)
set ratio 1.0;                                              # Ratio of DtAnalysis/dt (Need to set it manually)

# EXTRACT GROUND MOTION DATA --------------------------------------------------------
set data_fid [open $gmFileName "r"]
set data [read $data_fid]
close $data_fid
set data_new [split $data "\n"]
set timeData {}
set gmData {}
for {set k 0} {$k <= [expr [llength $data_new] - 2]} {incr k 1} {
    set data_t [lindex $data_new $k]
    lappend timeData [lindex $data_t 0]
    lappend gmData   [lindex $data_t 1]
}

set tMaxAnalysis [lindex $timeData end];                    # Maximum duration of GM analysis
set dt [expr [lindex $timeData 1] - [lindex $timeData 0]];  # ground motion sampling time
set dtAnalysis  [expr $dt*$ratio];                          # time-step for analysis

# INCLUDE DAMPING -------------------------------------------------------------------
# Only alphaM is needed since for SDOF,
set alphaM [expr $c/$m];
set betaK 0.;
set betaKinit 0.;
set betaKcomm 0.;
rayleigh $alphaM $betaK $betaKinit $betaKcomm; # RAYLEIGH damping

# DEFINE TIME SERIES AND LOAD PATTERN -----------------------------------------------
set loadTag 1;  # LoadTag for uniform ground motion excitation
set gmFact [expr $g*$gmFact]; # Since data in input file is in units of g.
set tsTag 1;
timeSeries Path $tsTag -dt $dt -values $gmData -factor $gmFact;
pattern UniformExcitation $loadTag $gmDirection -accel $tsTag;      # create Unifform excitation

# PERFORM DYNAMIC GROUND-MOTION ANALYSIS --------------------------------------------
set NewmarkGamma 0.50;  # Newmark-integrator gamma parameter (also HHT)
set NewmarkBeta  0.25;  # Newmark-integrator beta parameter

# CREATE THE SYSTEM OF EQUATIONS ----------------------------------------------------
system BandGeneral;

# CREATE THE CONSTRAINT HANDLER -----------------------------------------------------
constraints Plain;

# CREATE THE DOF NUMBERER -----------------------------------------------------------
numberer Plain;

# CREATE THE CONVERGENCE TEST -------------------------------------------------------
test NormUnbalance 1.0e-5 1000 0; # The norm of the displacement increment with a tolerance of 1e-5 and a max number of iterations of 1000. The 1/0 at the end
```

```
shows/doesn't show all iterations.

# CREATE SOLUTION ALGORITHM ------------------------------------------------------------
set algorithmBasic [split $algorithmString]
algorithm {*}$algorithmBasic

# CREATE THE INTEGRATION SCHEME --------------------------------------------------------
integrator Newmark $NewmarkGamma $NewmarkBeta;

# CREATE THE ANALYSIS OBJECT -----------------------------------------------------------
analysis $analysisType;

# RECORD AND SAVE OUTPUT ---------------------------------------------------------------
source generateRecorders.tcl; # Call file Recorders.tcl to record desired structural responses and save as an output file

# ANALYZE ------------------------------------------------------------------------------
set nSteps 1; # Set the number of steps in which the structure is to be analyzed. Here we go one step at a time
set tCurrent [getTime];
set ok 0;
while {$ok == 0 && $tCurrent <= $tMaxAnalysis} {
    set ok [analyze $nSteps $dtAnalysis];  # Analyze the structure for each time step. Sets ok to 0 if successful.
    if {$ok == 0} { puts " TIME: $tCurrent >> CONVERGED" }
    set tCurrent [getTime];
    # If the solution algorithm fails, as expected at reversals, change analysis option to Newton
    if {$ok != 0} {
        puts "Solution algorithm failed! Might be a load reversal! Changing algorithm to Newton"
        algorithm Newton;
        set ok [analyze $nSteps $dtAnalysis];
        # If successful, revert to original algorithm
        if {$ok == 0} { puts " TIME: $tCurrent >> CONVERGED" } {
            set tCurrent [getTime];
            puts "Changing to Newton helped. Going back to original algorithm"
            algorithm {*}$algorithmBasic
        }
    }
}
```