Command sequence in the main file

1. Create 1D transient reactive transport class object: `my_RT_trans`
2. Create 1D transient transport class object: `my_tpt_trans`
3. Create chemistry class object: `my_chem`
4. Write path to databases (including backslash or front slash depending on your OS)
5. Write path to input files for problem to be solved
6. Choose option for transport: either compute or read mixing ratios.
   If option=0 (ie. compute mixing ratios):
   - Read transport data: `my_tpt_trans` calls subroutine `initialise_transport_1D_transient_RT`
   - Allocate transport arrays: `my_tpt_trans` calls subroutines `allocate_arrays_PDE_1D` and `allocate_conc`
   - Compute mixing ratios: `my_tpt_trans` calls subroutine `compute_mixing_ratios_Delta_t_homog`
   - Set transport attribute in reactive transport object: `my_RT_trans` calls `set_transport_trans`
   - Read integration method for chemical reactions and set its attribute: `my_RT_trans` calls `set_int_method_chem_reacts`

   If option=1 (ie. read mixing ratios):
   - Set transport attribute in reactive transport object: `my_RT_trans` calls `set_transport_trans`
   - Read time discretisation: `my_RT_trans` calls `read_time_discretisation`
   - Read transport data to apply WMA: `my_RT_trans` calls `read_transport_data_WMA`
7. Read chemistry: `my_chem` calls `read_chemistry`
8. Call reactive mixing solver: there will be different solvers depending on the model for activity coefficients (ideal or not) and the WMA method (lumped or consistent): `my_chem` will call
   - If lumped and ideal: `solve_reactive_mixing_ideal_lump`
   - If consistent and ideal: `solve_reactive_mixing_ideal_cons`
   - If lumped and non-ideal: `solve_reactive_mixing_lump`
   - If consistent and non-ideal: `solve_reactive_mixing_cons`
9. Set chemistry attribute in reactive transport object: `my_RT_trans` calls `set_chemistry`
10. Write data and results: `my_RT_trans` calls `write_RT_1D`