Blake Hensel
Kyle Kent
Laura Jauch
Conner de la Cruz

Phase II: Design Plan

We used the state pattern to build this phase, with a state to represent each of the major user command options. Each state contains a prompt() function and a handle() function. The prompt() function initiates the command process for that class. The user response is filtered through main() which parses it and passes it on to the state's handle() method, which performs the desired action or calculation.

Main(): This is the class that starts the command process and also is responsible for exiting if, at any point, the user indicates they want to exit.

Initial(): Here the user is given the initial command options, to create or load a form. The state is then changed to Solver_Type() If input doesn't match one of the options, the question is asked again.

Solver_Type(): The user is asked whether they would like to solve a Stokes or Navier-Stokes equation. Their choice is passed on to Solver() in the form of a boolean.

Transition(): This contains the option list for user action. The user indicates if they would like to plot, refine, save, load or exit and the state is changed appropriately.

Solver(): This class forms the meat of the program. It is responsible for building the form. Here the user indicates relevant information to solving their equation: dimensions, number of elements, inflow and outflow conditions, etc. Once all the details have been specified, the form is solved based on its user indicated type. The user is notified of the degrees of freedom and energy error of the initial mesh. The form and relevant information is saved to the singleton form object so it can be referenced in other states. The user is then offered the option to change states. This class contains helper methods that perform parsing.

RefineS() and RefineNS(): These classes perform refining operations for Stokes and Navier-Stokes equations respectively. The user is given the option of p- or h- and auto or manual refinement. The input is parsed and the calculations performed. Then it prints the updated energy error, number of elements and degrees of freedom. The user is then prompted to change state.

Plot(): The user is given the option to plot u1, u2, pressure, stream function, mesh or error. A window pops up showing the indicated graph.

Save(): The user is prompted for a filename and then saves the form and relevant data under that filename. The state returns to Initial()

Load(): The user indicates the file from which they want to load. The form is restored and saved to the form object along with details about the polyOrder, type and re number.

Form(): This object class holds the saved form as well as some relevant information that needs to be stored/restored apart from the from itself. It contains methods for getting and setting the from and other data.

Instructions for use:

To start the program: python main.py
To build a form: ‘create’ and answer the questions.
To load a form: ‘load’ and indicate filename when prompted.
To exit: type ‘exit’ at any time to end the session.