**Adding new user input variables to DFNgen 2.0**

1. Add option/variable to an existing input file. Tag the option’s name with ‘:’ at the end. There must be at least 1 space or a new line in between the ‘:’ and the data.

E.g. newUserOption: 12

1. Add “extern *varType varName”* to input.h. Most user input variables are stored globally. Input.h must be included in any files that need access to them.
2. Update readInput.cpp. Declare the new global variable (the same variable as in step 1 but without the “extern” keyword) at the top of this file.

This file contains the function *getInput()*. This function is responsible for reading in user input files. *getInput()* needs to be updated to read in the new variable. I suggest looking for a similar variable, whether it be an array, a flag, or a number, and use that as an example to read in the new input option.

The function *searchVar*() is very helpful in reading variables from the user input file. The first argument is the file object (C++ ifstream object), the second argument is a string of the variable/option name in the input file including the ‘:’ at the end. After this function runs, the file pointer will be pointing to the data directly after the input options name (e.g. in step 1, the file pointer will be pointing to the white space directly after the colon.) All that is left is to read the input variable in to a c++ variable e.g. *file >> var.* NOTE: C++ is smart and will skip multiple spaces and/or new line characters.

If the option requires a list or array as the options parameters, see similar options in readInput.cpp. Instead of reading in directly to a variable (*file >> var)*, a function will be required to parse the list. See readInputFunctions.cpp and readInputFunctions.h for some examples on how to do this.

4. The last thing to do is to write/edit the code that will use the new option. Include input.h in any new file to access the global variable. If the new variable is an array, don’t forget to use *delete[]* to free its memory after the variable is no longer needed. If a new file was created, be sure to edit the makefile to include it in the built.