By completing the first two milestones, I learned how to use functions to further improve the modulization of the existing program from the last assignment. For example, clearKeyboard, pause, getInt, getIntInRange, and other helper functions were each declared and defined in respective contactHelper header and source files. These helper functions were subsequently utilized to reduce redundant coding for the repetitive tasks in Contact source file.

In programming, the function is used to complete a single specific task and remain separate from other functions. By dividing the program into small and manageable parts using functions, the user can easily maintain the modular design and improve the readability and maintainability of the program. Its portability is also greatly appreciated amongst the programmers. For example, they can use the predefined sets of functions in the libraries of the programming languages to perform useful jobs without additional coding.

In this exercise, the helper functions are stored in a different module from the contacts module. In this way, I was able to design a module that contains highly cohesive and lowly coupled tasks. For example, the contact module consists of the functions that manage the contact information from the user. In contrast, contactHelper module includes functions to mainly “help” the contact functions as they execute generic tasks such as getting inputs and checking the valid range of values.