Analysis

- **Introduction**
  - Student must design and make a program that is able to take an input as a list of integers and validate the input so that it can be used to calculate the total cost of the order by referencing the existing menu that has been created. The user should also be able to add, amend and delete menu items, and save menu changes that are saved to a file for later. The program must maintain a running total of order values, totals of quantity of each menu item ordered, loop the input for more orders and display order details for the user.

- **Key Requirements of the Program**
  - User should be able to input order details with table number and a string of numbers corresponding to menu items chosen
  - Program must be able to validate input data
  - Display the order details for printing
  - Loop for next order
  - Allow menu to be saved to a file
  - Allow for user to amend, add, delete menu items as well as save menu changes
  - Maintain running total of order values
  - Maintain running totals of the quantity
  - Save running totals to a file
  - Provide options to display the menu and running totals

- **What is decomposition?**
  - Decomposition is a general approach to solving a problem by breaking it up into smaller problems then solving it one problem at a time.
    - E.g. Separate function for validating input data
    - Separate function for calculating total
    - Separate function for writing to file and saving it

- **Sub-problems**
  - Sub-problem #1: Validating input data
    - The subprogram will take input data and then check the data to make sure that it fits the criteria of the input data. E.g. if input is wrong data type (e.g. string) the sub-program would not accept this input because then the whole program will not work with the data time.

  - Sub-problem #2: Calculating total cost of order
    - The subprogram will take the quantity of the menu items, then look at the menu items listed in the file and calculate the total cost by multiplying the amount of that item with the price listed in the menu item file.

  - Sub-problem #3: Saving and editing menu using files saved locally.
    - The sub-program will display a menu where the user view, edit and add to the menu items and add prices. The sub-program must be able to save changes to a file so that it can be accessed after the program has stopped running.

  - Sub-problem #4: Display final totals (e.g. **Balance due** and **Quantities** of menu items
    - The sub program will take the final totals from sub-problem #2 and it will display it in a generated table ready for printing.

  - I have broken down the problem into these specific sub-problems because each sub-problem represents a smaller problem that is crucial to the final program working. They are small and specific tasks that perform different and unique tasks from each other. Sub-problems should be of similar size, but of different processes/tasks than other sub-problems.