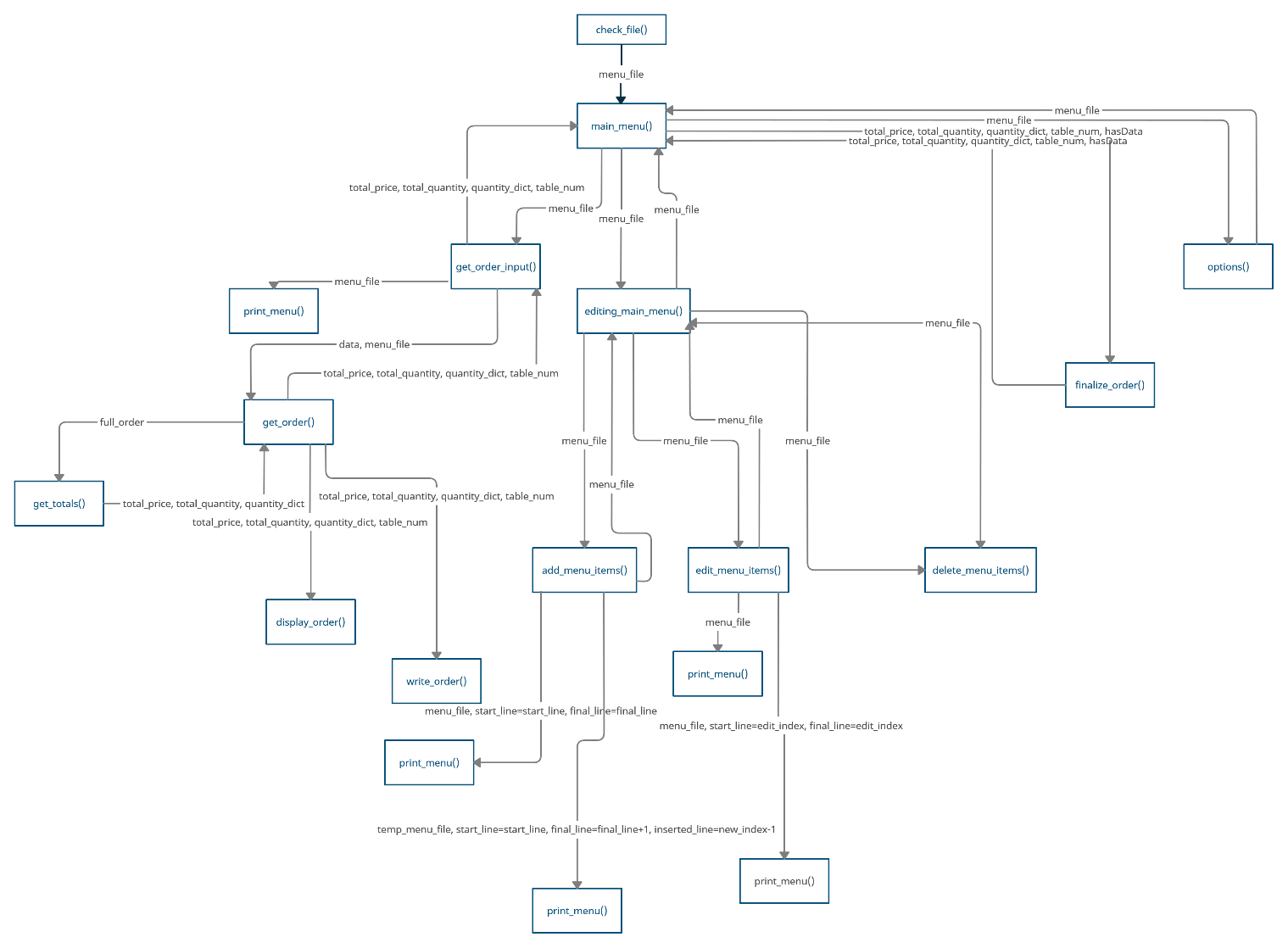
**Design**

* **Overall Plan**
  + This is a hierarchy chart displaying the whole program solution. When the program starts, check\_file() starts after main\_menu() is run and the user can choose either get\_order\_input(), editing\_main\_menu() finalize\_order() or options(). The subprograms that the user can choose all return information that is used with the other subprograms to work properly. Without this information the program will not work and information will not be passed around. These subprograms run and when the user decides to exit those sub programs they will be brought back to the main menu unless they choose to exit. This is a basic abstraction of how the overall program’s subprograms work together.

**Pseudo Code on next page.**



* **Individual subroutine plans**
  + Check “Pseudo Code.txt” for **FULL** Individual Subroutine plans in pseudo code
* **Subroutine Descriptions:**
  + ***finalize\_order()***– takes all the totals and data from what the user has entered as their order and then displays it for printing.
  + ***options()***– displays an options menu for the user so that they can choose if they want to manually reset the menu back to defaults
  + ***delete\_menu\_items()***– allows the user to delete existing menu items that are stored in the menu that will be eventually written to: “menu.txt”
  + ***write\_menu()***– takes the menu then, creates menu.txt and writes to it from menu\_file that is passed into it as an parameter
  + ***edit\_menu\_items()***– allows the user to take an existing menu item and edit it’s index, name, price or category and saves changes to the menu
  + ***add\_menu\_items()***– allows the user to add menu items to the menu. When adding an item, the user can decide it’s price, name, index number and category
  + ***editing\_main\_menu(****)* – Provides the menu interface for the user to choose to Add, edit or delete menu items in the menu. Exiting this menu takes the user to the main\_menu()
  + ***write*\_order()** – Takes the user’s order totals such as total price and total quantity and writes it to “running\_totals.txt”
  + ***display\_order()***– displays the current order with the quantities of each menu item ordered
  + ***get\_quantity()***– Generates a dictionary of how many of each item has been ordered in a simple dictionary format for display\_order()
  + ***get\_totals()*** – Calculates quantity and cost totals for display\_order()
  + ***get\_order()***– Checks order with menu in “menu.txt” and generates the variable: “full\_order”
  + ***print\_menu()***– Takes menu.txt from the “menu\_file” variable and prints it to the user when so that the user can see a clear view of the current menu with index numbers, prices, names and the category that it is in
  + ***get\_order\_input()***– Takes the user’s input for their order and validates it and formats it properly so that it can be used later in the program by other subroutines
  + ***main\_menu()***– The Main Menu of the program, it’s the first menu that the user sees. The user can choose to Input order data, edit menu items, finalize order, change options or quit to desktop.
  + ***check\_file()***– Checks for the menu.txt and creates menu.txt it if it doesn’t exist using the tuple: DEFAULT\_MENU
* **Test table for carrying out tests in the program:**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | **Stage 2 – Initial Test Plan** | **Stage 3 and 4 – Test Table** | |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test No.** | **Purpose of Test** | **Test Data** | **Expected Result** | **Actual Result** | **Action Needed/Comments** | | **1** | Check that menu index is only an integer | abc123.!@#!!|})(\*#& | “Invalid Index” | “Invalid Index” | No action needed | | **2** | Check that menu index is only an integer | 0 | “Out of range try again!” | “Out of range try again!” | No action needed | | **3** | Check that input order formats correctly | 6, 1, 8, 5,4 2, 3, 6 | *Shows correct output with items listed* | *Shows correct output with items listed* | No action needed | | **4** | Check if user can’t input a table number or menu index number higher than what is already accepted | 100,1,1,1,1000,200 | “We only have 10 tables! Table number must be lower than 10, please try again.” | “We only have 10 tables! Table number must be lower than 10, please try again.” | No action needed | | **5** | Check if user can’t input a category that doesn’t exist | Test123!1./.-123123ija | “Please enter a valid category, try again.” | “Please enter a valid category, try again.” | No action needed | | **6** | Check if user can’t input numeric characters as a name input for adding a menu item | 123 | “Name must contain alphabetic characters only, try again.” | “Name must contain alphabetic characters only, try again.” | No action needed | | **7** | Check if user can only input numeric characters as an input for the price of an item |  |  |  | No action needed | | **8** |  |  |  |  | No action needed | | **9** |  |  |  |  | No action needed | | **10** |  |  |  |  | No action needed | | **11** |  |  |  |  | No action needed | |