|  |  |
| --- | --- |
| **Requirements** | **Classification** |
| 1. Making sure that all necessary variables created and initialized | Functional |
| 1. Successful memory creation for the accumulator. In this project, we are creating an array of 1000 places for each memory location | Non-Functional |
| 1. BasicML program must be loaded into the main memory starting at location 00 | Non-Functional |
| 1. Display basic instructions about how to use the program to the user | Functional |
| 1. Get a number word input from the user by prompting | Functional |
| 1. Check if the number word contains at first digit a – or + sign and 4 digit number | Non-Functional |
| 1. If the user enters the word in incorrect format, have an exception function available to prompt the user again for correct formatting | Functional |
| 1. After successful user number input, prompt the user to enter a BasicML instruction. | Functional |
| 1. Check again if the instruction entered by the user is among the list of available instructions. If not, prompt the user again for correct instruction | Non-Functional |
| 1. While requesting data location for specific element, make sure that the location is not empty or occupied | Non-Functional |
| 1. Since we are using array as a memory container, we need an exception handler if the user enters a memory location less than 0 or greater than 1000. | Non-Functional |
| 1. Make sure to display all the necessary information to the user while running the program | Functional |
| 1. Successful Halt command to end the program | Non-Functional |
| 1. Display final information to the user | Functional |
| 1. Reset all the variables used and exit the program | Functional |