**SIC/XE Assembler Project**

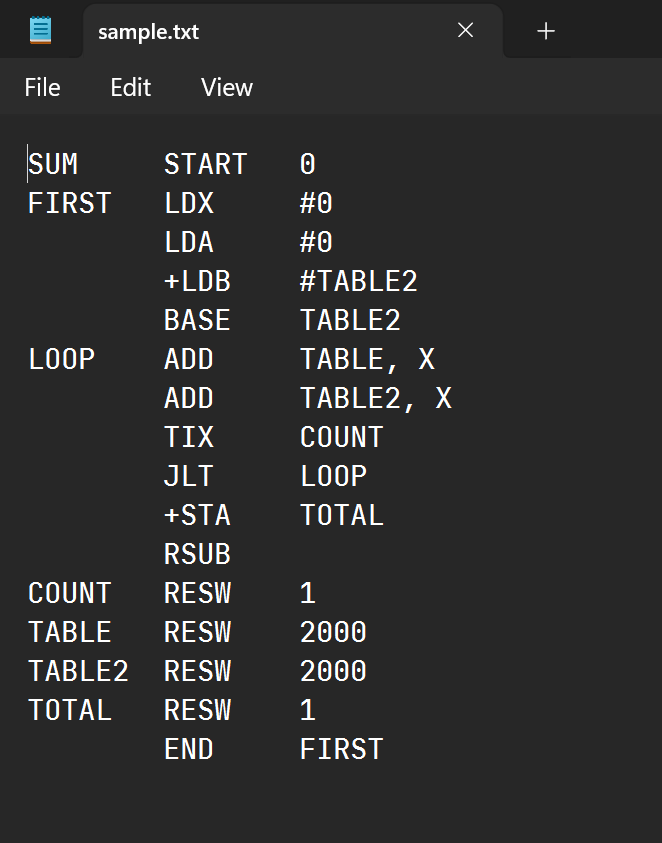
**CSN-252 Tutorial-8**

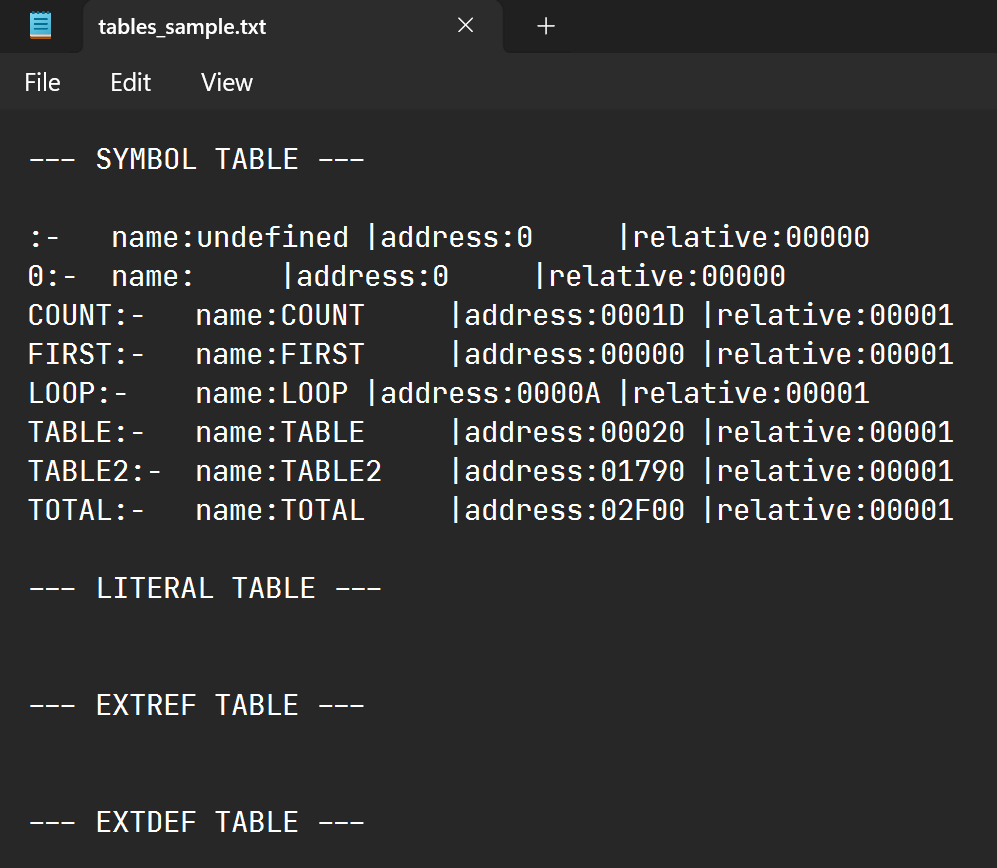
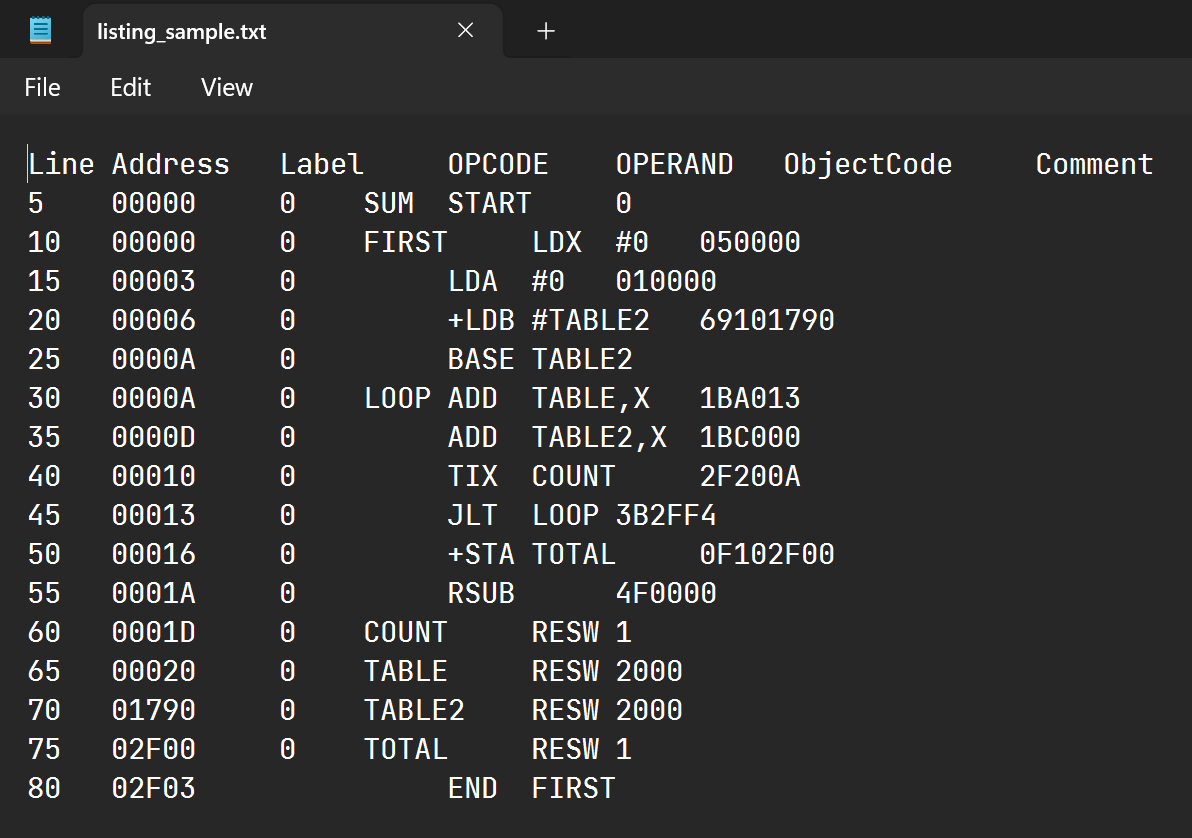
**My Enrolment No. is odd and hence, my assembler implements all the required SIC/XE functionalities with Control Section. The source code file contains pass1.cpp, pass2.cpp, tasks.cpp and tables.cpp.**

* **How to Run the Code**

1. Extract the **22114051.zip** file.
2. Open terminal in the **22114051** directory.
3. Compile the code using the command: **g++ pass2.cpp**, the compiled program will be written to **a.exe** or **a.out** depending on the operating system.
4. Run the compiled code using the command: **./a.exe** or .**/a.out**
5. Enter the assembly code file name in the prompt given on running the program.
6. The different results; intermediate, listing, object, tables, errors are written suffixed with the input file name.

* **Sample Code and Results**

**Sample Code 1 (sample.txt):** Question 3 of Section 2.2

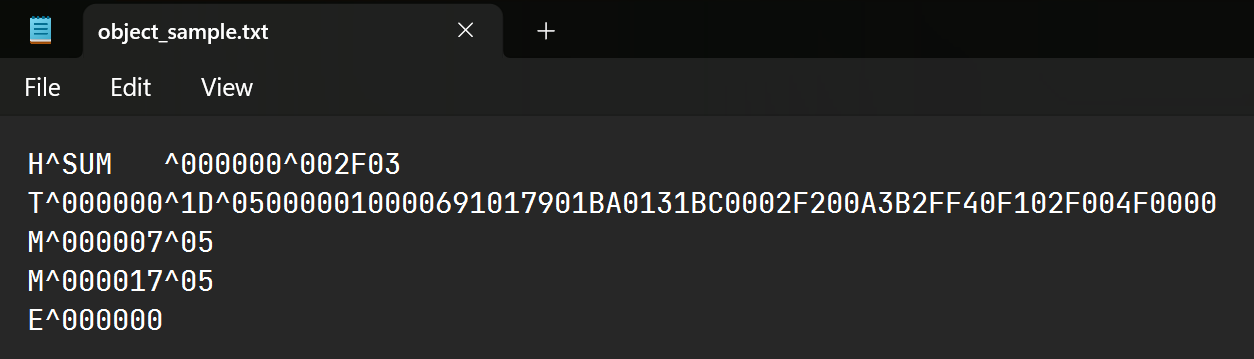
Assembled Results of Sample Code 1:

Graphical user interface, text

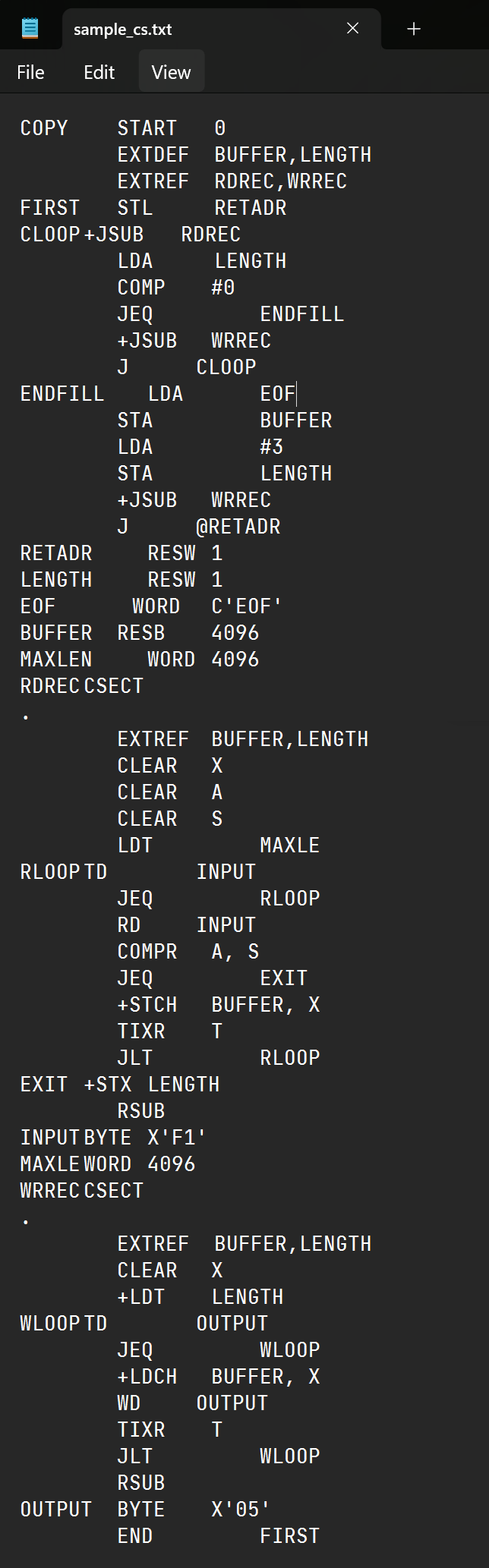
Description automatically generated

Graphical user interface, text, application

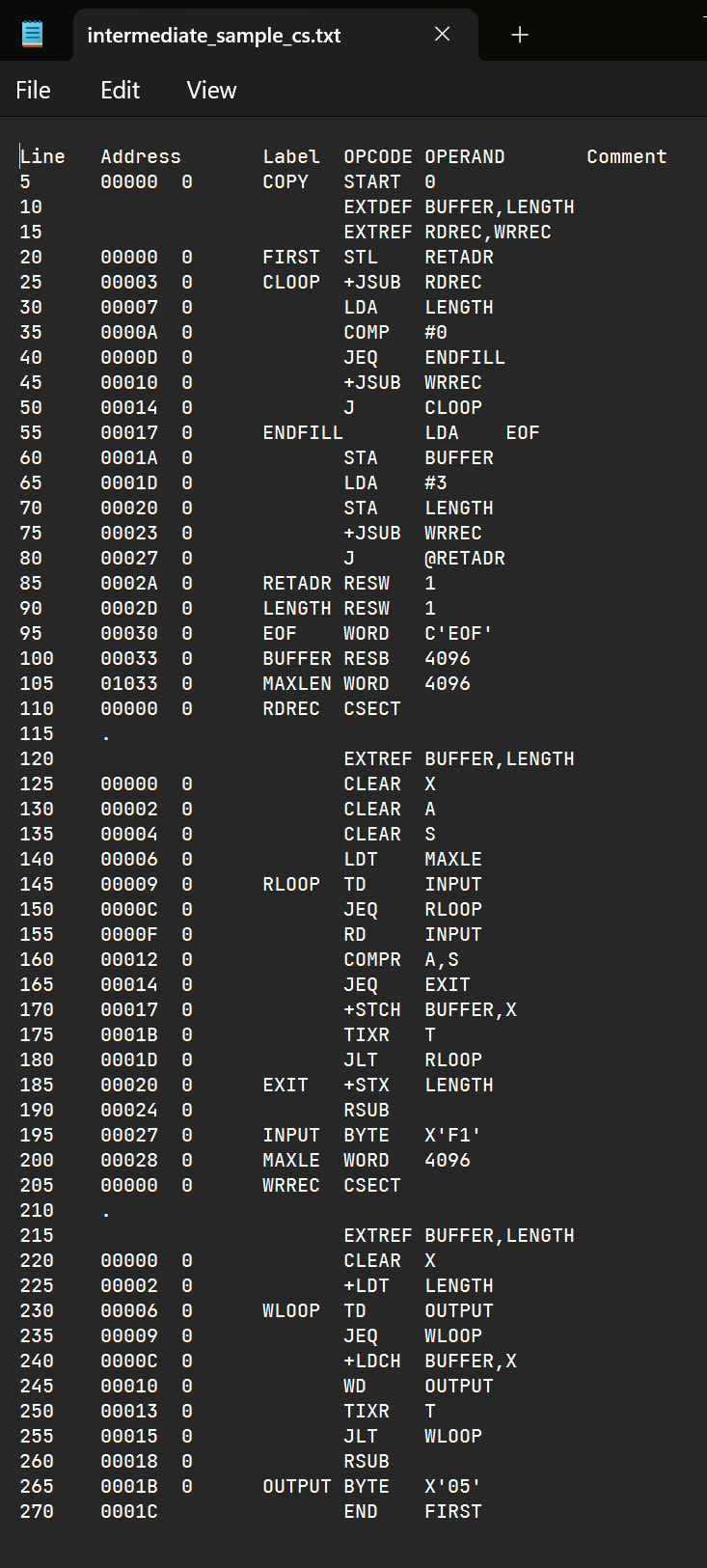
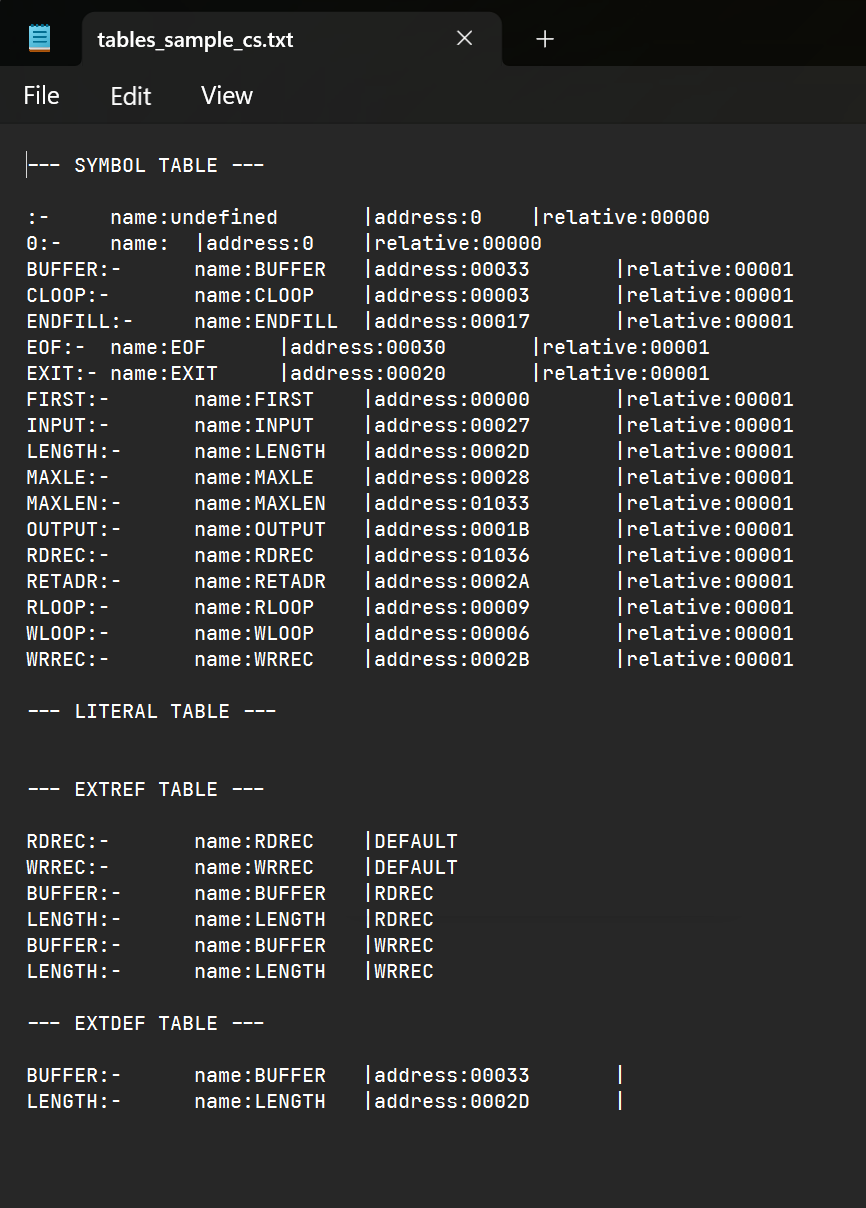
Description automatically generated with medium confidence

****

**Sample Code 2 (sample\_cs.txt):** Example Code with Control Section from LL Beck Textbook.



**Graphical user interface, application

Description automatically generated**Assembled Results of Sample Code 2:

* **Additional Notes**

1. Text

   Description automatically generatedGraphical user interface

   Description automatically generated with medium confidenceThe two sample inputs are provided in the folder with names sample.txt and sample\_cs.txt.
2. There are additional sample inputs sample\_literals and sample\_with\_error for testing with literals and for an incorrect assembly code provided respectively.
3. The results files are object\_program.txt, listing.txt, error.txt, tables.txt and intermediate.txt.
4. The README is provided both in .docx.q