APPENDIX-C

LABORATORY EXERCISES

This appendix will enable the reader

- To design and implement a lexical analyzer (*scanner*) for the given lexical specification of tokens of the language
- To design and implement the syntax analyzer (*parser*) for the given syntactic speciation of the structure of the language
- To generate the intermediate code on successful parsing of the source code
- To generate the machine code for a assumed target machine
- To do a mini project covering all the features of the un-optimized compiler for a restricted language

Hints:

1. Routines like Stack, First and Follow will appear in one place only.

2. All the exercises must be prepared with minimum three files:

   i. Header file.

   ii. Implementation file.

   iii. Application file where main function will be present.

   [For the convenience of display all the files are merged].

The idea behind using three files is to differentiate between the developer and user sides. In the developer side, all the three files could be made visible. For the user side only header file and application files could be made visible, which means that the object code of the implementation file could be given to the user along with the interface given in the header file, hiding the source file, if required.

3. It is recommended not to use any (I/O) functions (scanf, printf) in the implementation file since the user is not aware of the contents of the file. If any error message should be displayed to the user, error code can be returned from the implementation which can be mapped with the table of error message.

These hints may be followed when these exercises are made in deliverable form. However, for more clarity and better understanding, we have combined all the functions with appropriate printf functions.