**Probal Chandra Dhar  
Advanced Programming Languages**

**Project 2  
User's Manual**

**Setup and Compilation**

1. Download and unzip the submission from eLearning on a Linux box in the multi-platform lab.
2. The submission includes:

* main.c
* parser.c
* lexar.c
* symbol.c
* global.h
* stack.c
* error.c
* emitter.c
* makefile
* FunctionalDecompadvpl-p2.docx
* grammar-p2.docx
* userManual-p2.docx
* legal1.txt
* legal2.txt
* legal3.txt
* legal4.txt
* illegal1.txt
* illegal2.txt
* illegal3.txt
* illegal4.txt
* script

1. Environment: This program has been tested in the multi-platform lab and will run there.
2. Compiling. This program includes a Makefile. At the command line in Linux, type make. The program produces an executable entitled compiler

**Running the program:** Be sure the sample programs (legal1.txt, legal2.txt, legal3.txt, legal4.txt, illegal1.txt, illegal2.txt, illegal3.txt, illegal4.txt) are in the same directory as the executable. Issue the command ./compiler <filename>. Filename is required to run this program because input is reading from files.

By doing this you can run one program at a time. If you want to run 8 programs at a time you can run the script by issuing the command ./script. Please make sure you’ve to run the Makefile before this command. Type make to run Makefile.

If the syntax of the code in the file match with the provided grammar, then it’ll generate a file same name with the input file but with a .dat extension. For example, if this command

./compiler legal1.txt

given after the make command then it’ll generate output like this:

code generated in legal1.dat file

then to see the code you can use cat command.

cat legal1.dat

It’ll show the generated code in the commandline interface.

Script is a bash script that will run the program 8 times with different sample program I’ve added in the folder.

If an output file already exists in the folder then every new run will overwrite the previous file with the newer code.

**User input:** no user interaction with the program is required.

**Output:** all output goes to the console and every successful compilation will generate a file same with the given filename but with a different extension .dat. Output will be the similar to this:

code generated in legal1.dat file

code generated in legal2.dat file

code generated in legal3.dat file

code generated in legal4.dat file

line 4: syntax error factor

line 8: syntax error factor

line 2: wrong identifier name

line 1: syntax error match

and it’ll create four files in the folder.

legal1.dat

legal2.dat

legal3.dat

legal4.dat