Mitchell Nguyen
CS 357
4 December 2017

Programming Project – Summary

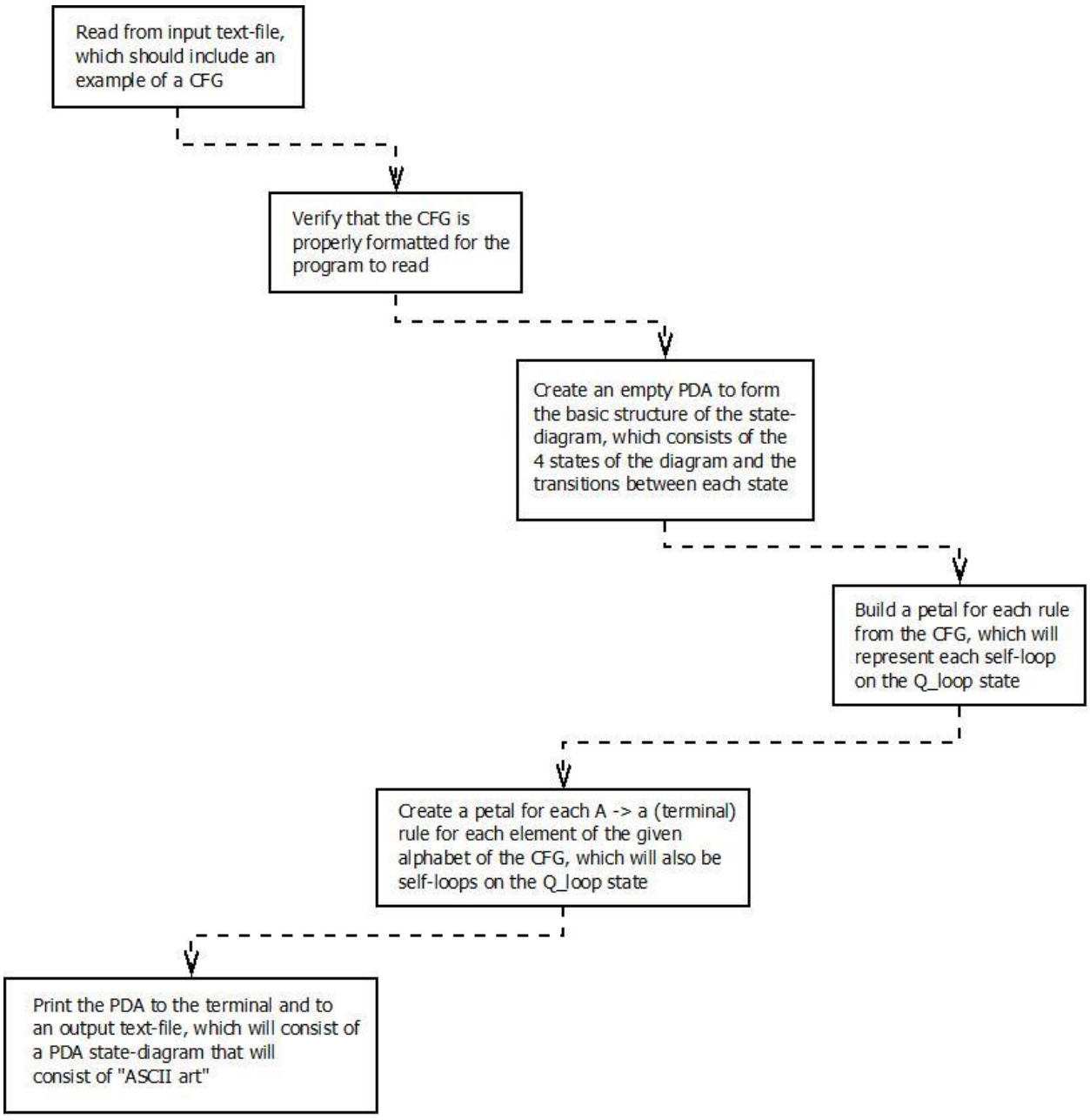
a.) Name: Mitchell Nguyen

b.) How to compile and run the program:

a. For this assignment, I used **MobaXTerm** to compile and execute the C-code that I created.
b. Enter into the directory that includes the input text-files and C-code with the "cd"-commands.
c. Type in the following command (to compile the program):
    i. **gcc -o proj nguyenmi19_cs357_programmingProject.c**
d. Type in the following command (to run the program using data from a certain input file (for the '*' character, please use either '1', '2', or '3' as provided)):
    i. **./proj input_*.txt**
e. After performing these commands, the resulting PDA should be displayed on the **terminal** and printed out in a text file called **output.txt**

(… Project Summary continues onto next page…)

c.) Overall design of the project (follows data structures provided in project proposal document):



(… Project Summary continues onto next page…)

d.) Results of testing:

a. The resulting PDA that comes from the CFG data in **input_1.txt** is the same PDA of **test case #1** that is provided in the project proposal document (NOTE: the project proposal's PDA has some minor mistakes on the last two "petals" of Q_loop, such that the order of the transitions should be switched on both petals).
b. The resulting PDA that comes from the CFG data in **input_2.txt** is the same PDA of **test case #2** that is provided in the project proposal document
c. The resulting PDA that comes from the CFG data in **input_3.txt** is portrayed as expected (the data of output.txt matches with my hand-drawn version of the PDA).

d. Limitations of input text-file format:
    i. Do not include spacebar characters (' ') in the input text-file.
    ii. Instead of utilizing the 'Ɛ' character in the input text-files, I had to substitute '3' as the empty character since MobaXTerm could not read a text-file with Unicode in it. MobaXTerm only wanted the standard ANSI format of all the characters in the input text-file.
    iii. The input text-file consists of the alphabet{a,b}, such that the only terminals that should exist in the input CFG's are 'a' and 'b'.
    iv. I have restricted my program to only accept the input text-files under the names **input_1.txt**, **input_2.txt**, and **input_3.txt**.
    v. My code only runs on rules that have at most three variables and/or terminals on the right-hand side of the arrow. For example, the following rules are permitted in my program:
        1. S->ABC
        2. S->abB
        3. S->a
    vi. … But my program does not process over rules that have more than three variables and/or terminals on the right-hand side of the arrow, such as:
        1. S->ABCd
        2. S->abCDE

e. Further note:
    i. My program opens and writes to one output file called **output.txt**. It will have the PDA of the input text-file that was most recently executed on (since running the program on some input text-file simply replaces the text in the output text-file).