COMP 451

FORMAN CHRISTIAN COLLEGE

(A CHARTERED UNIVERSITY)

COMPILER CONSTRUCTION

Class Project

Fall 2022

It's an open books and open notes task. Use of Internet is allowed. This programming task should be done in complete isolation. No group formation is allowed. You CANNOT share your code with any other student studying the same course. Any such attempt will be dealt with seriously.

Grading Criteria

Working Code: 80%

Properly formatted Report: 20%

Important: You need to submit a well formatted and well written report for this assignment. The report should carry following sections:

- Introduction about the problem in hand.
- Your short code segments followed by a detailed description explaining how you built up the logic of the given code snippet.
- If you have used some user defined functions in your code make sure to given a detailed description about their working.
- Additional functionalities and / or exclusions (if any) should be stated with separate heading in bold face font.
- Code description should be augmented (if possible) by the screen shots of the output for that piece of code to make your description clearer and more concise.
- Start early. **NO** additional time in any case what so ever will be granted.
- *Your code will be checked for plagiarism on Turnitin. All students who will have a similarity index larger than 60% will be called for a detailed face to face Viva.*

Hard Deadline: Submit your code file on Moodle class page on or before Tuesday Jan 24, 2023 11:59 pm. Bring the hardcopy of report for the project with you on your final exam day.

Make sure to create a zip file of your submission and give it a name using given format:

COMP451_<Your RollNumber>_classProject

For example, COMP451_12345678_classProject

Submissions through email will NOT be considered for grading.

COMP 451

Project Task - 1 [40 Marks]

In this task you need to write a code in C that will implement the functionality of “Stack Implementation Table” for the LR(0) parser.

Here is the grammar that you will use:

E → BB

B → cB

B → d

Some of the possible valid input strings that we can generate from this grammar are:

{cdd, ccdd, cccdd, ccccdd, cccccdd . . .}

Your task is to follow all the steps which include,

- Augmenting the grammar.
- Drawing the LR(0) item sets

for the given grammar.

Now you will list down the parse table for the LR(0) parser.

Note that all the above steps should be completed using **paper and pencil**. Make sure to incorporate these in your report as well.

Now that we have listed down the parse table for the parser, you need to write a program using C language. Your program should accept a string on command line argument and list down the complete stack implementation table on screen for the LR(0) parser showing whether the string is accepted or rejected.

Note that you must incorporate appropriate checks in your program. Some of these may be:

- Check that user must provide $ symbol in her input.
- The input string must consist of alphabets only. No numeric or special character (except one $ symbol at the end of the string) are allowed.

Once you are done with the program test it using different strings and choose at least three valid and three invalid inputs for your program. Include the screen shots of the output in your report.

A sample run is provided for your convenience.

COMP 451

Project Task - 2 [40 Marks]

In this task you need to write a code in C that will implement the functionality of “Stack Implementation Table” for the LL(1) parser.

Here is the grammar that you will use:

S → Aa

A → BD

B → b | ε

D → d | ε

Some of the possible valid input strings that we can generate from this grammar are:

{a, ba, da, bda}

Your task is to follow all the steps which include,

- Listing down the FIRST and FLLOW sets of all the non-terminal symbols in the grammar.
- List down the LL(1) parsing table.

Note that all the above steps should be completed using **paper and pencil**. Make sure to incorporate these in your report as well.

Now that we have listed down the parse table for the parser, you need to write a program using C language. Your program should accept a string on command line argument and list down the complete stack implementation table on screen for the LL(1) parser showing whether the string is accepted or rejected.

Note that you must incorporate appropriate checks in your program. Some of these may be:

- Check that user must provide $ symbol in her input.
- The input string must consist of alphabets only. No numeric or special character (except one $ symbol at the end of the string) are allowed.

Once you are done with the program test it using different strings and choose at least three valid and three invalid inputs for your program. Include the screen shots of the output in your report.

A sample run is provided for your convenience.

For LR(0) parser:

```
$ gcc lr.c -o lr
$ ./lr ccdd$
```

The output is as shown below:

Stack	Input	Action
$0	ccdd$	s3
$0c3	cdd$	s3
$0c3c3	dd$	s4
$0c3c3d4	d$	r3
$0c3c3B6	d$	r2
$0c3B6	d$	r2
$0B2	d$	s4
$0B2d4	$	r3
$0B2B5	$	r1
$0E1	$	Accept

For LL(1) parser:

```
$ gcc ll.c -o ll
$ ./ll bda$
```

The output is as shown below:

Stack-1	Stack-2	Action
S$	bda$	S → Aa
Aa$	bda$	A → BD
BDa$	bda$	B → b
bDa$	bda$	
Da$	da$	D → d
da$	da$	
a$	a$	
$	$	Accepted