
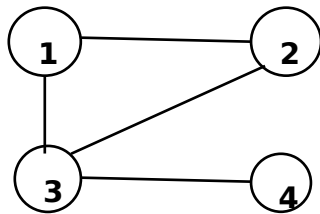


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Compiler Construction	Course Code:	
	Degree Program:	BS (CS)	Semester:	Spring 2020
	Exam Duration:	3 hrs & 15 min	Total Marks:	100
	Paper Date:	23-Jun-2020	Weight	45%
	Section:	ALL	Page(s):	
	Exam Type:	Final (Online)		

An undirected graph can be represented in many ways. One common method is to use an adjacency matrix; another is to use an adjacency list. Moreover we can use XML-like language for this purpose. Consider the following graph for example:



We can represent this graph as follows:

```
<graph>
  <node> 1, 2, 3, 4 </node>
  <edge>
    (1,2), (1,3), (2,3), (3,4)
  </edge>
</graph>
```

The same graph can be represented using an adjacency matrix as follows:

	1	2	3	4
1		1	1	
2	1		1	
3	1	1		1

4			1	
---	--	--	---	--

You are required to develop a compiler in C/C++ or Java, to convert the XML-like representation of any given (undirected) graph into an equivalent adjacency matrix representation.

Following are the exam questions and distribution of marks:

1. List down all the tokens. Give regular definition for any complex token. A complex token is the one that has more-than-one lexemes. (10 marks)
2. Write a (left-recursive) CFG for the compiler. (10 marks)
3. Give (C/C++ or Java) source code of the complete translator. (80 marks)

Submission instructions:

Each student shall submit two files: one pdf and one cpp. The pdf file shall contain scanned images of the hand-written answers to the first two questions. While the cpp file shall contain the complete source code for the translator. Use your roll no to name the files. For example, if the roll no is 12-6789, the names of the file shall be 12-6789.pdf and 12-6789.cpp.

Your code should follow the design. You are not allowed to use any powerful libraries or tools for scanning or parsing. However, you can use the standard libraries, such as C++ string library.

For the lexical analyzer, you are not required to follow the method involving DFA's; rather you can develop as you like. However, the parser shall be written using the recursive-descent parsing covered in the class.

Plagiarism shall not be tolerated. Minimum penalty would be an F grade in the course.

I shall build executable using a standard C++/Java compiler. Your program shall take an input file (in.txt), and shall display the output on screen (no output file). Assume the input file is placed in the same directory where your executable shall run. So use simple file handling; no need to use command-line arguments.

Your code should be executable preferably on Windows.