COMP421 Compiler Design Project

100 points possible (10% of final grade)

Assigned Date: 13/11/2012

Due Date: **11/01/2013 by 10:00am – strict deadline**

1 Project Objective and Description

In this project, you will gain experience using the lex and yacc tools. To be more specific, you will build an interpreter for *UnicTextLang*. It is recommended to study the lex and yacc mini tutorials as well as the examples provided on the class web site.

2 *UnicTextLang*: A Simple Text Formatting Language

Your interpreter will generate a text document (*UnicText.txt*) that is formatted according to the specifications in a source text file.

2.1.1 Short Description of Language

All programs in this language have to start with **begin{document}** and end with the **end{document}**. All document properties must appear right after the **begin{document}**. If any property is missing, then an error is generated. The main text properties (except the **begin{document}** and **end{document}**) could appear in any order and frequency in the document. A new document is generated that conforms to the formatting specification as stated in the source program.

2.1.2 Document Properties

Construct	**Description**	**Error Handling**
title{string}	Specifies the title of the document, which is a string. It must be centered in the generated document, based on the line width.	Lexical errors
author{string}	Specifies the author of the document, which is a string. It must be centered in the generated document, based on the line width.	Lexical errors
date{day/month/year}	Specifies the date. All arguments are integers. It needs to be a valid date. It must be centered in the generated document, based on the line width.	Lexical errors and semantic errors for the date.

\pagesetup{lines_per_page, line_width}	Specifies the page setup in terms of lines per page and line width. Both arguments are integers. It could be assumed that the width of the character is fixed.	Lexical errors and semantic errors on the datatype of the arguments.
\tabsize(size**)**	Specifies the tab size, which is integer.	Lexical errors and semantic errors on the value of size.

2.1.3 Main Text Properties

Construct	**Description**	**Error Handling**
\begin{document}	Start of the document	Lexical errors
\end{document}	End of the document	Lexical errors
\section{string}	Indicates the start of a new section. The title will be capitalized in the generated document. One empty line before and after the title will be generated as well. The section numbering starts at 1.	Lexical errors
\paragraph{string}	Indicates the start of a paragraph. The first sentence is indented according to the tab size.	Lexical errors
\newline	Generates an empty line.	Lexical errors

\begin{itemize}
\item{string}
\item{string}
…
\end{itemize}
	Creates a bulleted list (using the * character), with a tab between the character and the item string.	Lexical errors

\begin{enumerate}
\item{string}
\item{string}
…
\end{enumerate}
	Creates an enumerated list, with a tab between the character and the item string. The numbering starts at 1, and gets reset every time a new enumerated list is created.	Lexical errors

2.1.4 An Example Source Program and its Corresponding Generated Text

```
\begin{document}

\pagesetup{30, 100}
\tabsize(5)
\title{Why I love Compiler Design}
\author{COMP421 Student}
\date{07/11/2012}

\section{What is Compiler Design}
\paragraph{Compiler Design is the study of designing and implementing
compilers for languages. It is…}
\paragraph{The compiler phases are grouped into front-end phases and back-end
phases. The front-end of the compiler consists of the following phases:}
\begin{itemize}
\item{Lexical Analysis}
\item{Syntax Analysis}
\item{Semantic Analysis}
\end{itemize}

\newline
\newline
\paragraph{This is the end of the first section.}

\section{Why it is my best course}
\paragraph{It is my best course because:}
\begin{enumerate}
\item{Learn a Lot}
\item{Fun}
\item{Challenging project}
\end{enumerate}


\end{document}
```

Why I love Compiler Design
By
COMP421 Student
07/11/2012

1. WHAT IS COMPILER DESIGN

Compiler Design is the study of designing and implementing compilers for languages. It is…

The compiler phases are grouped into front-end phases and back-end phases. The front-end of the compiler consists of the following phases:

*   Lexical Analysis
*   Syntax Analysis
*   Semantic Analysis

This is the end of the first section.

2. WHY IT IS MY BEST COURSE

It is my best course because:

1.   Learn a Lot
2.   Fun
3.   Challenging Project

-1-

Note: the characters per page is supposed to be 100, the tab size is 5.
At the end of each page (after 30 lines), you will add an extra line that will serve as the footer of the document, which keeps track of the page number in the generated document.

3 Helpful Hints

It is recommended to follow the steps outlined below:

- Specify the lexical part of the language in *UnicTextLang.l* . Check if the tokens are properly identified before adding the parsing functionality.
- Define the grammar of *UnicTextLang*. At the same time, determine global variables. Add parsing functionality, one feature at a time.

4 Additional Features for Teams with 3 Members

If your team has 3 members, then you will also be required to implement the following feature (20% of the total project grade):

- Add a flag feature. Multiple flags can be defined, and used in nested **if**s. In the case below, the flag called debug is set to true, thus the itemized list will be produced. Otherwise, if it was set to false, the enumerated list will be generated.

```
\begin{document}

\pagesetup{30, 100}
\tabsize(5)
\title{Why I love Compiler Design}
\author{COMP411 Student}
\date{07/11/2010}

\flag(debug){true}

\section{What is Compiler Design}
\paragraph{Compiler Design is the study of designing and implementing
compilers for languages. It is…}
\paragraph{The compiler phases are grouped into front-end phases and back-end
phases. The front-end of the compiler consists of the following phases:}
if (debug) then
{
\begin{itemize}
\item{Lexical Analysis}
\item{Syntax Analysis}
\item{Semantic Analysis}
\end{itemize}
}
else
{
\begin{enumerate}
\item{Lexical Analysis}
\item{Syntax Analysis}
\item{Semantic Analysis}
\end{enumerate}
}

\end{document}
```

5 Fewer Features for Teams with 1 Member

If your team has 1 member, then you do not need to add page numbers in the generated document.

6 Assignment Deliverables

You must electronically submit a tar (or zip) file containing the following:

- Source code of the project
  - ✓ *UnicTextLang.l*
  - ✓ *UnicTextLang.y*
  - ✓ *UnicTextLang.c*
  - ✓ Supporting header files you created for *UnicTextLang*
  - ✓ 3 source programs that you used to test your interpreter (e.g. *source1.txt, source2.txt, source3.txt*) and the corresponding generated document for each source.
- README file
  - ✓ This is a plain text file that contains information about the program (purpose, programming language, operating system that can be used to run your program), the programmer(s), and *detailed instructions on how to compile and run it*.
- Report document
  - ✓ BNF grammar of *UnicTextLang* language
  - ✓ Testing cases: 3 source programs that you used to test your interpreter and the generated documents.
  - ✓ A few sentences describing any problems or challenges that you faced for this assignment as well as *any incomplete functionality*.

7 Grading Criteria

A program that does not compile and run on the platform indicated in your README file will get no credit. In addition, a README file without compile and run instructions will also get no credit. The overall project worths 100 points and they are distributed as follows:

- *UnicTextLang* implemented and its interpreter functions as described **(80 points)**
- Report and README file include all the required information **(20 points)**

8 Submission Guidelines

You must electronically submit your project via email (dionysiou.i@unic.ac.cy) no later than **10:00am on11/01/2013**. Use the subject line **COMP421Project**. If you don't receive an acknowledgment email message within 24 hours of your submission, then that means I haven't received your project. **<u>You are responsible to check that your project was submitted successfully!</u>**