COMP 3031 Assignment 2
Flex and Bison Programming

Fall 2019

Due 5pm, 8th Nov. 2019 Friday

1. Problem Description

In this assignment, you will implement a parser for a Mini-Markdown language by filling in the blanks in the provided flex and bison code skeletons.

The Mini-Markdown language is used to describe an HTML page. The BNF grammar rules for the language are divided into two sets, with one set for the bison program and the other for Flex:

Bison Rules: //<mddoc> is the starting symbol of the language

```
<mddoc> ::= <empty> | <mddoc> <paragraph>

<paragraph> ::= <newline> | <pcontent> <newline>

<pcontent> ::= <header> | <rftext>

<header> ::= <headermark> <blanks> <rftext>

<rftext> ::= <rftext> <blanks> <rftextword> | <rftext> <rftextword> | <rftextword>

<rftextword> ::= <textnum> | <image> | <format>

<image> ::= ![<text>](<text>=<int>@<int>)

<format> ::= **<text>** | _<text>_ | **<format>** |_<format>_

<text> ::= <text><blanks><textnum> | <text><textnum> | <textnum>

<textnum> ::= <textword> | <int>
```

Flex Rules:

```
<textword> ::= <textword> <char> | <char>

<char> ::= a|b...|z|A|B...|Z|:|/|.|-|,|’

<headermark> ::= # | ## | ### | #### | ##### | ######

<int> ::= <int> <digit> | <digit>

<digit> ::= 0|1|2|3|4|5|6|7|8|9

<newline> ::= <newlinechars> | <blanks> <newlinechars>

<newlinechars> ::= <newlinechars> \n | \n

<blanks> ::= <blanks> <blankchar> | <blankchar>

<blankchar> ::=  \t | “ ”
```

Your Mini-Markdown parser will parse an input Mini-Markdown text file and output an HTML file. For each line separated by newline characters in the Mini-Markdown file, a pair of " " and " " tags will be added to surround the line content in the output HTML file. Similarly, the header and formatting information in the Mini-Markdown are also translated into HTML tags. We give examples and list provided helper functions for these translations in the following table:

Mini-Markdown	HTML tag	Helper function
This is a line.	This is a line.	generate_paragraph(const char* text)
some text	some text	generate_bold(const char* text)
some text	some text	generate_italic(const char* text)
![imgName](imgUrl=wid@hei)	<img src="imgUrl" alt="imgName" width="wid" height="hei">	generate_image(const char* imgName, const char* imgUrl, int width,int height)
# some text ## some text ### some text #### some text ##### some text ###### some text	some text some text some text some text some text some text	generate_header(int level, const char* text)

2. Your work

We provide a zip package containing the following files:

assignment2.pdf	Assignment 2 description
global.h, global.c	Helper functions
Makefile	Makefile that supports "make" and "make clean"
miniMD2html.l	Flex code skeleton for you to fill in
miniMD2html.y	Bison code skeleton for you to fill in
miniMD2html-example	An example executable parser
test_input1.md, test_output1.html	Example input file and corresponding output file

Your tasks are:

1. Add missing flex definitions and rules in ***miniMD2html.l***
2. Add missing tokens and grammar rules in ***miniMD2html.y***

We have marked the blanks that you can fill in with comment lines:

```
/************* Start: ... */

/* End: ...*/
```

Note: Do **not** modify the provided files *Makefile, global.h* and *global.c.*

3. Compilation and Testing

You can run the given example binary parser on the given example input file:

```
./miniMD2html-example test_input1.md
```

You can compile your code on a CS Lab 2 machine with the command "`make`". After successful compilation, you can run the binary on the given example input file, and compare your output with the given output file on the example:

```
./miniMD2html test_input1.md > myoutput.html

diff test_output1.html myoutput.html
```

Your code should give exactly the same output as our solution code gives; otherwise, you may lose marks.

4. Submission:

- Zip your two source files, ***`miniMD2html.l`*** and ***`miniMD2html.y`***, into a single package using exactly the following command (case-sensitive):

  ```
  zip miniMD2html.zip miniMD2html.l miniMD2html.y
  ```

- Submit your zipped file ***`miniMD2html.zip`*** to Canvas.
- **No late submission will be accepted.**
- Your submission will be compiled and run on a CS Lab 2 machine. If it cannot be compiled or run properly, you may get 0 marks for this assignment.