GLASS

Milestone One Progress Report

Team: Tommy Galletta, Alexander Lockard

Faculty Advisor: Dr. Stansifer

Now with a fancy new logo!

Milestone Two Task Matrix

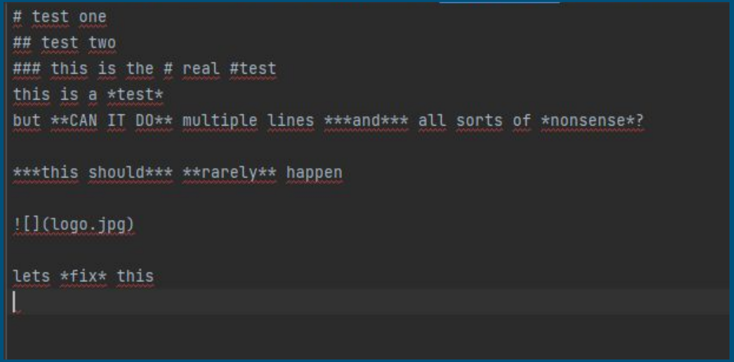
Task	Completion %	Tommy	Xander	Todo
Parser generator intermediary checkpoint	100%	80%	20%	
Investigate other parser generators	70%	35%	35%	Keep investigating
Solidify syntax specification format “version one”	50%	25%	25%	Ideas in place, need to make grammar
Implement, test, and demo XML output	100%	10%	90%	

Task Discussion

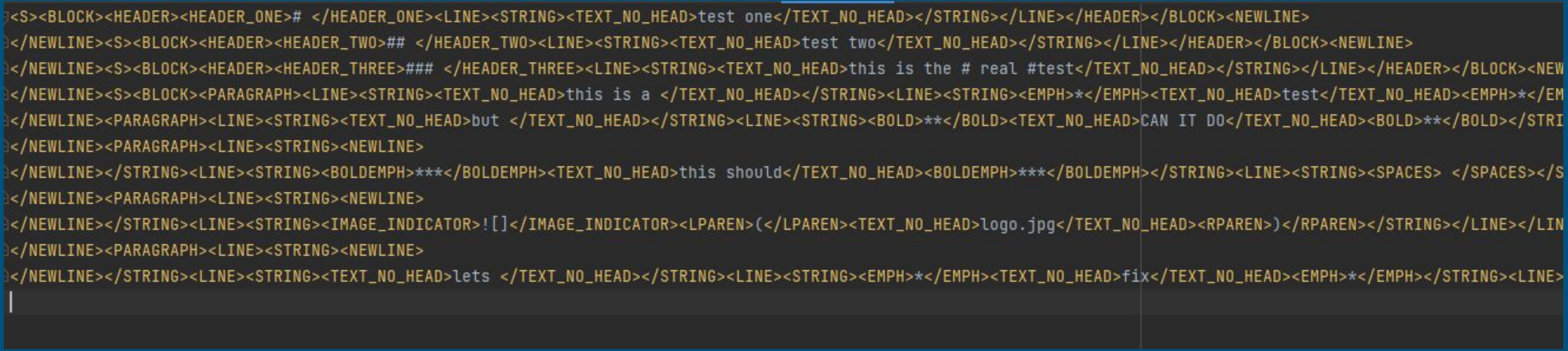
Parser Generator Development

- It kind of works!
- Implementation of First, Closure, and Transition functions
- Grammar is hard coded for now but LR(1) parsing is working

Task Discussion (continued)



```
# test one
## test two
### this is the # real #test
this is a *test*
but **CAN IT DO** multiple lines ***and*** all sorts of *nonsense*?

***this should*** **rarely** happen

![](logo.jpg)

lets *fix* this
```

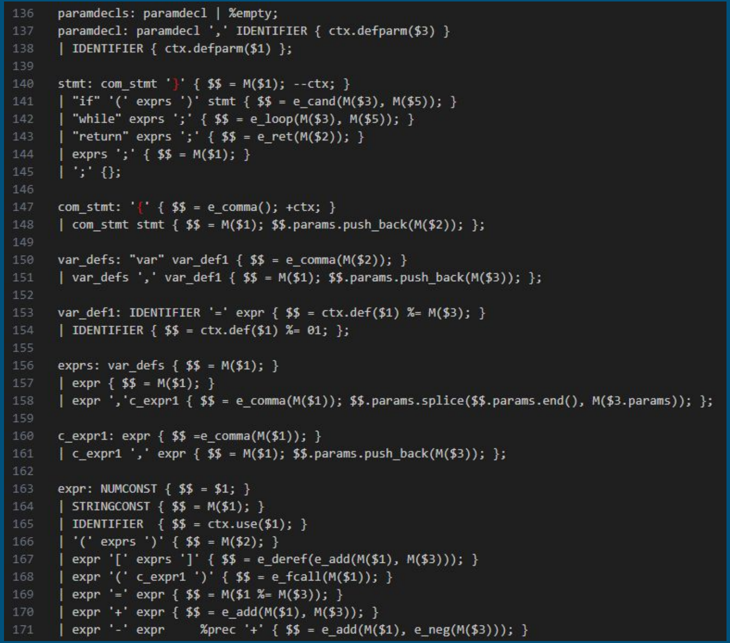


```
<S><BLOCK><HEADER><HEADER_ONE># </HEADER_ONE><LINE><STRING><TEXT_NO_HEAD>test one</TEXT_NO_HEAD></STRING></LINE></HEADER></BLOCK><NEWLINE>
</NEWLINE><S><BLOCK><HEADER><HEADER_TWO>## </HEADER_TWO><LINE><STRING><TEXT_NO_HEAD>test two</TEXT_NO_HEAD></STRING></LINE></HEADER></BLOCK><NEWLINE>
</NEWLINE><S><BLOCK><HEADER><HEADER_THREE>### </HEADER_THREE><LINE><STRING><TEXT_NO_HEAD>this is the # real #test</TEXT_NO_HEAD></STRING></LINE></HEADER></BLOCK><NEW
</NEWLINE><S><BLOCK><PARAGRAPH><LINE><STRING><TEXT_NO_HEAD>this is a </TEXT_NO_HEAD></STRING><LINE><STRING><EMPH>*</EMPH><TEXT_NO_HEAD>test</TEXT_NO_HEAD><EMPH>*</EM
</NEWLINE><PARAGRAPH><LINE><STRING><TEXT_NO_HEAD>but </TEXT_NO_HEAD></STRING><LINE><STRING><BOLD>**</BOLD><TEXT_NO_HEAD>CAN IT DO</TEXT_NO_HEAD><BOLD>**</BOLD></STRI
</NEWLINE><PARAGRAPH><LINE><STRING><NEWLINE>
</NEWLINE></STRING><LINE><STRING><BOLDEMPH>***</BOLDEMPH><TEXT_NO_HEAD>this should</TEXT_NO_HEAD><BOLDEMPH>***</BOLDEMPH></STRING><LINE><STRING><SPACES> </SPACES></S
</NEWLINE><PARAGRAPH><LINE><STRING><NEWLINE>
</NEWLINE></STRING><LINE><STRING><IMAGE_INDICATOR>![]</IMAGE_INDICATOR><LPAREN>(</LPAREN><TEXT_NO_HEAD>logo.jpg</TEXT_NO_HEAD><RPAREN>)</RPAREN></STRING></LINE></LIN
</NEWLINE><PARAGRAPH><LINE><STRING><NEWLINE>
</NEWLINE></STRING><LINE><STRING><TEXT_NO_HEAD>lets </TEXT_NO_HEAD></STRING><LINE><STRING><EMPH>*</EMPH><TEXT_NO_HEAD>fix</TEXT_NO_HEAD><EMPH>*</EMPH></STRING><LINE>
```

Task Discussion (continued)

Investigating other parser generators

- Investigation of tree-sitter complete
- Investigation of yacc/bison complete
- Still would like to investigate 2 more.

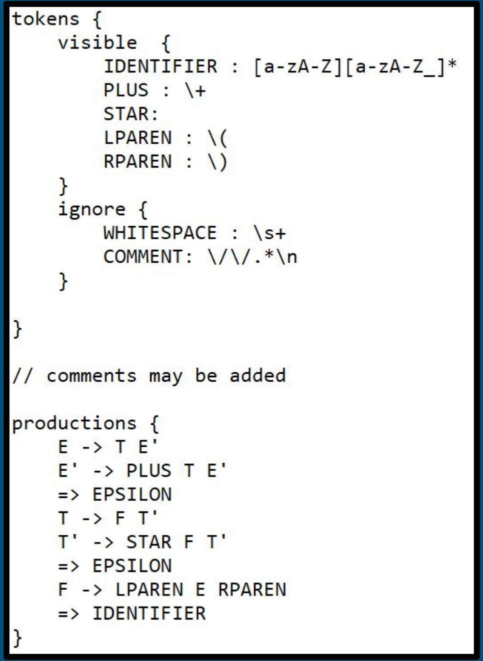


```
paramdecls: paramdecl | %empty;
paramdecl: paramdecl ',' IDENTIFIER { ctx.defparm($3) }
| IDENTIFIER { ctx.defparm($1) };

stmt: com_stmt '}' { $$ = M($1); --ctx; }
| "if" '(' exprs ')' stmt { $$ = e_cand(M($3), M($5)); }
| "while" exprs ';' { $$ = e_loop(M($3), M($5)); }
| "return" exprs ';' { $$ = e_ret(M($2)); }
| exprs ';' { $$ = M($1); }
| ';' {};

com_stmt: '{' { $$ = e_comma(); +ctx; }
| com_stmt stmt { $$ = M($1); $$.params.push_back(M($2)); };

var_defs: "var" var_def1 { $$ = e_comma(M($2)); }
| var_defs ',' var_def1 { $$ = M($1); $$.params.push_back(M($3)); };

var_def1: IDENTIFIER '=' expr { $$ = ctx.def($1) %= M($3); }
| IDENTIFIER { $$ = ctx.def($1) %= 01; };

exprs: var_defs { $$ = M($1); }
| expr { $$ = M($1); }
| expr ','c_expr1 { $$ = e_comma(M($1)); $$.params.splice($$.params.end(), M($3.params)); };

c_expr1: expr { $$ =e_comma(M($1)); }
| c_expr1 ',' expr { $$ = M($1); $$.params.push_back(M($3)); };

expr: NUMCONST { $$ = $1; }
| STRINGCONST { $$ = M($1); }
| IDENTIFIER  { $$ = ctx.use($1); }
| '(' exprs ')' { $$ = M($2); }
| expr '[' exprs ']' { $$ = e_deref(e_add(M($1), M($3))); }
| expr '(' c_expr1 ')' { $$ = e_fcall(M($1)); }
| expr '=' expr { $$ = M($1 %= M($3)); }
| expr '+' expr { $$ = e_add(M($1), M($3)); }
| expr '-' expr     %prec '+' { $$ = e_add(M($1), e_neg(M($3))); }
```

Task Discussion (continued)

Solidify syntax specification format

- Ideas are laid out but work needs to be continued here.
- Grammar for the format needs to be solidified



```
tokens {
    visible  {
        IDENTIFIER : [a-zA-Z][a-zA-Z_]*
        PLUS : \+
        STAR:
        LPAREN : \(
        RPAREN : \)
    }
    ignore {
        WHITESPACE : \s+
        COMMENT: \/\/.*\n
    }

}

// comments may be added

productions {
    E -> T E'
    E' -> PLUS T E'
    => EPSILON
    T -> F T'
    T' -> STAR F T'
    => EPSILON
    F -> LPAREN E RPAREN
    => IDENTIFIER
}
```

Task Discussion (continued)

Implement, test, and demo XML output

- XML parsing kind of works.
- Some complications have appeared with the library we are using so we have decided to write our own.

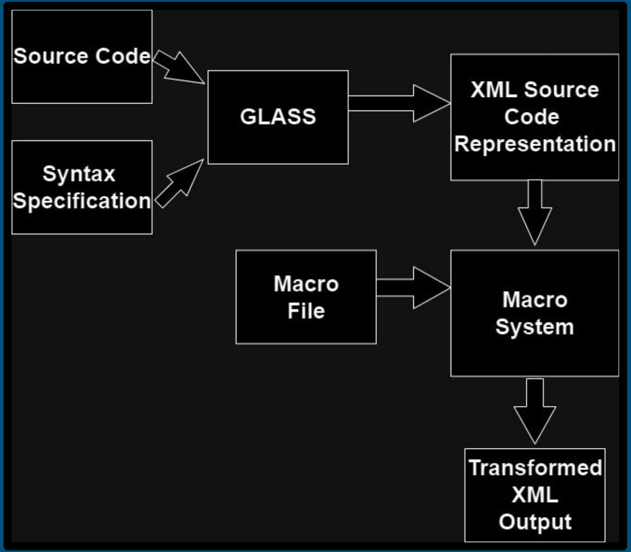
Milestone Three Plan

Task	Tommy	Xander
Syntax specification file reading	Implement "version one" syntax specification reading	Test syntax specification reading
Basic macro interpretation / XML manipulation	Test and debug macro interpreter	Implement basic macro interpreter
Continued research of parser generators	Investigate 1-2 parser generator tools	Investigate 1-2 parser generator tools
Begin documentation	Documentation for parser generator system	Documentation for XML macro system

Discussion of Planned Tasks

Syntax specification file reading

- By the end of Milestone 3, we hope to have most of our main “pipeline” in place for our tool, so that future milestones can be focused on each of us and communicating with the user more effectively.
- Based on a specification we decide, a user should be able to define a grammar and have that grammar be read in for parsing by the parser generator.
- Once a parse table is built from the user’s grammar, the user should be able to input an input file and have it be parsed by the tool.



Discussion of Planned Tasks (continued)

Basic macro interpretation/XML manipulation

- User should be able to input a macro file along with the XML generated from the parser generator tool.
- The macro file will be interpreted and the specified operations will be performed on the XML file.

Discussion of Planned Tasks (continued)

Continued research of parser generators

- We want our tool to be easier to understand, quicker to learn, and more “beginner friendly” than the other options.
- In order to make the best tool we need a better understanding of our market share.

Discussion of Planned Tasks (continued)

Begin Documentation

- We want our tool to have extensive documentation
- We believe documentation should be started sooner rather than later to avoid it being rushed.

Faculty Advisor Feedback

- Advisor is pleased with current state of parser generator
- Advisor is fine with state of XML output.
- Advisor does not see XML as the best form of output but understands we need some form of medium representation.
- Advisor hopes to see parser conflict resolution to be added

Questions?