**CS 421 Project Part B - Parser for Japanese**

* **Must email factored rules to me before working on the parser. Due Week 12 Monday 4/15**
* **Preliminary executable is due Week 14 Monday 4/29**
* **All required files are in ParserFiles on Empress. You must use them to be ready for Part C.**
* **Look for @@@@ for common mistakes.**

**Empress:**

**/cs/cs421LK/CS421Progs/ParserFiles**

**What to do in parser.cpp:**

**1) Update next\_token() and match() which are found week9A notes.**

**Update next\_token()** to save the string returned from the scanner

in globally accessible **saved\_lexeme** whenever **saved\_token** is updated.

You will need it for producing error messages.

**Please use these variable names and make sure they are “globally” accessible because you will need them in Part C.**

**Update match()** to display the matched token\_type when it succeeds. (used **for tracing the program \*\***)

**e.g. "Matched WORD1"**

**2) Write the parser functions for the following BNF rules.**

**Note that you have to apply left-factoring to some of the rules.**

**\* Coding style should be exactly the same as in Week 8 Notes to receive credit!**

**\* At the very start of each function,**

**cout the function name (**used **for tracing the program \*\*)**

**e.g. "Processing <story>"**

**e.g. "Processing <s>"**

**Rules in BNF:**

All non-terminals are in <>

Optional parts are in []

Repeatable (zero or more) parts are in {}

1 <story> ::= <s> { <s> } // stay in the loop as long as a possible start

// of <s> is the next\_token (note it can be CONNECTOR or WORD1 or PRONOUN)

2 <s> ::= [CONNECTOR] <noun> SUBJECT <verb> <tense> PERIOD

3 <s> ::= [CONNECTOR] <noun> SUBJECT <noun> <be> PERIOD

4 <s> ::= [CONNECTOR] <noun> SUBJECT <noun> DESTINATION <verb> <tense> PERIOD

5 <s> ::= [CONNECTOR] <noun> SUBJECT <noun> OBJECT <verb> <tense> PERIOD

6 <s> ::= [CONNECTOR] <noun> SUBJECT <noun> OBJECT <noun> DESTINATION

<verb> <tense> PERIOD

**// Refer to Left Factoring file to make these into**

// one rule until things start to differ.

7 <noun> ::= WORD1 | PRONOUN

8 <verb> ::= WORD2

9 <be> ::= IS | WAS

10 <tense> := VERBPAST | VERBPASTNEG | VERB | VERBNEG

**Additional Submission due on/before Week 12 Monday 4/15 (see the file on Left Factoring ):**

**Hint on Left Factoring\_w12A.docx**

You must email me the new set of S rules after you have applied left factoring completely. **You must use descriptive non-terminal names so that error messages will make sense.**

You cannot start coding until this is approved by me.

**Everyone should be involved in this task.**

**Syntax Error Handling:**

**Generate errors messages using the given format/phrases. Do not deviate. @@@@**

**match fails** => call **syntax\_error1** and generate

"SYNTAX ERROR: expected ***token\_type*** but found ***saved\_lexeme*** and halt. \*\* exit(1)

**e.g. SYNTAX ERROR: expected WORD2 but found asa**

**switch default** in a parser function => call **syntax\_error2** and generate

"SYNTAX ERROR: unexpected ***saved\_lexeme*** found in ***parser function***" and halt. \*\* exit(1)

**e.g. SYNTAX ERROR: unexpected nai found in tense**

**Make sure lexical errors are still generated before syntax error messages.**

**Possible extra credit features to include:**

* **Syntax error correction** for syntax\_error1:
* Interact with the user to correct syntax errors (skip token/replace token)

In this case, error messages must also be sent to **errors.txt** so that the user can

study them later.

* **Ability to turn on and off** the tracing messages (see **\*\*** above).

Syntax\_error1 – skip token or assume correct token was there. See OutputEC.txt.

Syntax\_error2 – exit(1).

**How to Start Parsing:**

**New Test Driver: main() – do nothing else!!!!**

- opens the input file (name obtained from the user)

- calls the <story> to start parsing

**<the old driver main() in the scanner should be removed>**

**Required Comments: @@@@**

**\* Each function should be commented by the corresponding grammar rule.**

**\* Each function should be commented by the programmer's name.**

**Required Testing:**

**Test cases are available in partBtest# (1-6). Read the comments in these files.**

**Model output for 2 cases are available. Test 7 is for EC.**

**Must divide the Work as follows:**

* **Member A: the two syntax error functions**
* **Member B: match() and next\_token().**
* **Member C: main()**

**Then, divide the non-terminals equally to code. Everyone should code non-terminals.**

**Everyone must work together and help debug each other’s code.**

**Everyone must meet periodically to put together the work and help debug each other’s code.**

**If a member is failing to do his/her work, other members should take over the task to complete the program so that it can be used for the next part of the project.**

**You will get to report who should get the credit for the task in a file called ParserWork.txt.**

**What to Submit on/before Week 14 Monday 4/29:**

**Please */cs/submitIt\_LK* your parser executable (.out created on empress @@@@)**

**Your executable must be named group#parser.out (e.g. group3parser.out)**

**Please output your group # and group members at the start of the program.**

**Please submitIt parserWork.txt with the requested info.**

**[5pts toward the project] for contributing members only if both the left factored rules and these files were received before the due dates and corrected right away as requested.**

**Your left factored rules will go into your project report, so keep it!!!**

**End of Part B.**