CS421 Final Project: Compiler

Group 6

Chris Childers

Ly Dung

Lenson Paulo Laca

**Section 0: State of the Program**

The program is working perfectly…….

Extra Credit Features:

* Enabling/Disabling Trace Messages

How it works: When program is ran the user is given an option for turning on or off the trace message.

Saving Error messages into a text file

How it works: This feature is integrated into the program when an error occurs it is simultaneously displayed in the screen and written into a text file “error.txt”.

**Section 1: DFA**

****

**Section 2: Scanner code that match your DFAs**

**#include<iostream>**

**#include <stdio.h>**

**#include<fstream>**

**#include<string>**

**#include<cstdlib>**

**#include <ctype.h>**

**using namespace std;**

**//Token type for the word in the language**

**enum tokentype {CONNECTOR, DESTINATION, ERROR, EOFM, IS, OBJECT, PERIOD, PRONOUN, SUBJECT, WAS, WORD1, WORD2, VERB, VERBNEG, VERBPAST, VERBPASTNEG};**

**int scanner(tokentype & , string & );// Looking for the token type for the word in the language**

**bool mytoken(string );// Does the word belong to the language?**

**bool myPeriod(string );// Does the word is period?**

**string convertE(string);// Convert japaneses to english**

**//Purpose: convert japaneses to english**

**//Algorithm: Looking for correspond english word**

**// Parameter:String**

**//Return: string**

**//Author: Dung Ly**

**string convertE(string a){**

**if(a=="watashi"){**

**a="I/me";**

**}**

**else if(a=="anata"){**

**a="you";**

**}**

**else if(a=="kanojo"){**

**a="she/her";**

**}**

**else if(a=="sore"){**

**a="mata";**

**}**

**else if(a=="mata"){**

**a="Also";**

**}**

**else if(a=="soshite"){**

**a="Then";**

**}**

**else if(a=="shikashi"){**

**a="However";**

**}**

**else if(a=="dakara"){**

**a="Therefore";**

**}**

**return a;**

**}**

**//Purpose: Convert tokentype to string**

**//Algorithm: set the similar string as token type**

**//Parameter: token type**

**//Return: string**

**//Author: Dung Ly**

**string convertT(tokentype t){**

**string a;// Return string for token type**

**switch(t){**

**case CONNECTOR:**

**a="CONNECTOR";**

**break;**

**case DESTINATION:**

**a="DESTINATION";**

**break;**

**case ERROR:**

**a="ERROR";**

**break;**

**case EOFM:**

**a="EOFM";**

**break;**

**case IS:**

**a="IS";**

**break;**

**case OBJECT:**

**a="OBJECT";**

**break;**

**case PERIOD:**

**a="PERIOD";**

**break;**

**case PRONOUN:**

**a="PRONOUN";**

**break;**

**case SUBJECT:**

**a="SUBJECT";**

**break;**

**case WAS:**

**a="WAS";**

**break;**

**case WORD1:**

**a="WORD1";**

**break;**

**case WORD2:**

**a="WORD2";**

**break;**

**case VERB:**

**a="VERB";**

**break;**

**case VERBNEG:**

**a="VERBNEG";**

**break;**

**case VERBPAST:**

**a="VERBPAST";**

**break;**

**case VERBPASTNEG:**

**a="VERBPASTNEG";**

**break;**

**default:**

**a="Nothing";**

**break;**

**}**

**return a;**

**}**

**//Purpose: Looking for token type for a input word**

**//Algorithm: Calling mytoken function and period function to determined the tokentype of the word**

**//Parameter: tokentype , string**

**//Return: int**

**//Author: Dung Ly**

**int scanner(tokentype & a, string & w)**

**{**

**bool test1=mytoken(w);// Calling token type function**

**// the the word is period => return token type**

**if(myPeriod(w)==true){**

**a=PERIOD;**

**return 1;**

**}**

**// the word belongs to the language=> find it token type**

**else if(test1==true)**

**{**

**// Checking for word 1 and word 2 1st**

**if(w[w.length()-1]=='I'||w[w.length()-1]=='E')**

**{**

**a=WORD2;**

**}**

**else**

**{**

**a=WORD1;**

**}**

**//check if the word belongs to a reservered word list or not. Override the result of a.**

**if (w=="masu")**

**a= VERB;**

**if(w=="masen")**

**a=VERBNEG;**

**if(w=="mashita")**

**a=VERBPAST;**

**if(w=="masendeshita")**

**a= VERBPASTNEG;**

**if(w=="desu")**

**a= IS;**

**if(w== "deshita")**

**a= WAS;**

**if(w== "o")**

**a= OBJECT;**

**if(w== "wa")**

**a=SUBJECT;**

**if(w=="ni")**

**a=DESTINATION;**

**if(w=="watashi")**

**a=PRONOUN;**

**if(w=="anata")**

**a=PRONOUN;**

**if(w=="kare")**

**a=PRONOUN;**

**if(w=="kanojo")**

**a=PRONOUN;**

**if(w=="sore")**

**a=PRONOUN;**

**if(w== "mata")**

**a=CONNECTOR;**

**if(w== "soshite")**

**a=CONNECTOR;**

**if(w== "shikashi")**

**a= CONNECTOR;**

**if(w=="dakara")**

**a=CONNECTOR;**

**if(w== "eofm")**

**a=EOFM;**

**//================================**

**return 1;**

**}**

**// The word is not in the langaue => token type is error**

**else if (test1==false)**

**{**

**a=ERROR;**

**cout<<"Lexical Error: "<< w <<" is not valid token"<<endl;**

**return 1;**

**}**

**return 0;**

**}//the end**

**//Purpose: Check if the word belogn to the language or not**

**//Algorithm: Using the language DPA**

**//Parameter: string**

**//Return: bool**

**//Author: Dung Ly**

**bool mytoken(string s){**

**int flag=0;// flag to stop checking if the state is found**

**int state = 0;// state the current state**

**int charpos = 0;// character position**

**// if the word is eofm => end of find return true**

**if(s=="eofm")**

**return true;**

**// Going in to the loop**

**while (s[charpos] != '\0')**

**{**

**if(s[s.length()-1]=='I'||s[s.length()-1]=='E')**

**s[s.length()-1]=tolower(s[s.length()-1]);**

**bool list1=s[charpos]=='a'||s[charpos]=='e'||s[charpos]=='i'||s[charpos]=='o'||s[charpos]=='u';**

**bool list2=s[charpos]=='b'||s[charpos]=='g'||s[charpos]=='h'||s[charpos]=='k'||s[charpos]=='m'||s[charpos]=='n'||s[charpos]=='p'||s[charpos]=='r';**

**bool list3=s[charpos]=='b'||s[charpos]=='g'||s[charpos]=='h'||s[charpos]=='k'||s[charpos]=='m'||s[charpos]=='p'||s[charpos]=='r';**

**bool list4=s[charpos]=='d'||s[charpos]=='y'||s[charpos]=='j'||s[charpos]=='w'||s[charpos]=='z';**

**bool list5=s[charpos]=='b'||s[charpos]=='g'||s[charpos]=='h'||s[charpos]=='k'||s[charpos]=='m'||s[charpos]=='p'||s[charpos]=='r'||s[charpos]=='d'||s[charpos]=='y'|s[charpos]=='z'||s[charpos]=='w';**

**//==========From state 0===============**

**if(flag==0&&state==0&&s[charpos]=='c')**

**{**

**state=5; // new state**

**flag=1;// stop go to any other if's**

**}**

**else if(flag==0&&state==0&&s[charpos]=='s')**

**{**

**state=3;**

**flag=1;**

**}**

**else if(flag==0&&state==0&&s[charpos]=='t')**

**{**

**state=6;**

**flag=1;**

**}**

**else if(flag==0&&state==0&&list1)**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==0&&list2)**

**{**

**state=2;**

**flag=1;**

**}**

**else if(flag==0&&state==0&&list4)**

**{**

**state=4;**

**flag=1;**

**}**

**//============From state 1============**

**if(flag==0&&state==1&&s[charpos]=='n')**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==1&&s[charpos]=='s')**

**{**

**state=3;**

**flag=1;**

**}**

**else if(flag==0&&state==1&&s[charpos]=='c')**

**{**

**state=5;**

**flag=1;**

**}**

**else if(flag==0&&state==1&&s[charpos]=='t')**

**{**

**state=6;**

**flag=1;**

**}**

**else if(flag==0&&state==1&& list5 )**

**{**

**state=2;**

**flag=1;**

**}**

**else if(flag==0&&state==1&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**//==========From State 2============**

**if(flag==0&&state==2&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==2&&s[charpos]=='y')**

**{**

**state=4;**

**flag=1;**

**}**

**//=========From State 3===========**

**if(flag==0&&state==3&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==3&&s[charpos]=='h')**

**{**

**state=4;**

**flag=1;**

**}**

**//=========From State 4===========**

**if(flag==0&&state==4&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**//=========From State 5===========**

**if(flag==0&&state==5&& s[charpos]=='h')**

**{**

**state=4;**

**flag=1;**

**}**

**//=========From State 6===========**

**if(flag==0&&state==6&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==6&&s[charpos]=='s')**

**{**

**state=4;**

**flag=1;**

**}**

**//=========From State 7===========**

**if(flag==0&&state==7&& s[charpos]=='s')**

**{**

**state=3;**

**flag=1;**

**}**

**else if(flag==0&&state==7&& s[charpos]=='t')**

**{**

**state=6;**

**flag=1;**

**}**

**else if(flag==0&&state==7&& s[charpos]=='c')**

**{**

**state=5;**

**flag=1;**

**}**

**else if(flag==0&&state==7&& list2)**

**{**

**state=2;**

**flag=1;**

**}**

**else if(flag==0&&state==7&& list1)**

**{**

**state=1;**

**flag=1;**

**}**

**else if(flag==0&&state==7&& list4)**

**{**

**state=4;**

**flag=1;**

**}**

**if(flag==0)**

**{**

**return false;**

**}**

**flag=0;// reset the flag**

**charpos++;//move to new character in the word**

**}//end of while**

**if (state == 0||state==1) return(true); // end in a final state**

**else return(false);**

**}**

**//Purpose:**

**//Algorithm:**

**//Parameter:**

**//Return:**

**//Author:**

**bool myPeriod(string s){**

**int state = 0;**

**int charpos = 0;**

**while (s[charpos] != '\0') {**

**if(state==0 && s[charpos]=='.')**

**state=1;**

**else if (state==1)**

**return false;**

**charpos++;**

**}**

**if(state ==1) return (true);**

**else return (false);**

**}**

**Section 3: Scanner test results**

**Test1:**

**Enter Input file: test.txt**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :rika**

**The type: ISThe word :desu**

**The type: PERIODThe word :.**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :sensei**

**The type: ISThe word :desu**

**The type: PERIODThe word :.**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :ryouri**

**The type: OBJECTThe word :o**

**The type: WORD2The word :yarI**

**The type: VERBThe word :masu**

**The type: PERIODThe word :.**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :gohan**

**The type: OBJECTThe word :o**

**The type: WORD1The word :seito**

**The type: DESTINATIONThe word :ni**

**The type: WORD2The word :agE**

**The type: VERBPASTThe word :mashita**

**The type: PERIODThe word :.**

**The type: CONNECTORThe word :shikashi**

**The type: WORD1The word :seito**

**The type: SUBJECTThe word :wa**

**The type: WORD2The word :yorokobI**

**The type: VERBPASTNEGThe word :masendeshita**

**The type: PERIODThe word :.**

**The type: CONNECTORThe word :dakara**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :kanashii**

**The type: WASThe word :deshita**

**The type: PERIODThe word :.**

**The type: CONNECTORThe word :soshite**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD1The word :toire**

**The type: DESTINATIONThe word :ni**

**The type: WORD2The word :ikI**

**The type: VERBPASTThe word :mashita**

**The type: PERIODThe word :.**

**The type: PRONOUNThe word :watashi**

**The type: SUBJECTThe word :wa**

**The type: WORD2The word :nakI**

**The type: VERBPASTThe word :mashita**

**The type: PERIODThe word :.**

**The type: EOFMThe word :eofm**

**Test2:**

**Enter Input file: test.txt**

**The type: WORD1The word :daigaku**

**==========Lexical Error: college is not valid token=====================**

**The type: ERRORThe word :college**

**The type: WORD1The word :kurasu**

**==========Lexical Error: class is not valid token=====================**

**The type: ERRORThe word :class**

**The type: WORD1The word :hon**

**==========Lexical Error: book is not valid token=====================**

**The type: ERRORThe word :book**

**The type: WORD1The word :tesuto**

**==========Lexical Error: test is not valid token=====================**

**The type: ERRORThe word :test**

**The type: WORD1The word :ie**

**==========Lexical Error: home\* is not valid token=====================**

**The type: ERRORThe word :home\***

**The type: WORD1The word :isu**

**==========Lexical Error: chair is not valid token=====================**

**The type: ERRORThe word :chair**

**The type: WORD1The word :seito**

**==========Lexical Error: student is not valid token=====================**

**The type: ERRORThe word :student**

**The type: WORD1The word :sensei**

**==========Lexical Error: teacher is not valid token=====================**

**The type: ERRORThe word :teacher**

**The type: WORD1The word :tomodachi**

**==========Lexical Error: friend is not valid token=====================**

**The type: ERRORThe word :friend**

**The type: WORD1The word :jidoosha**

**==========Lexical Error: car is not valid token=====================**

**The type: ERRORThe word :car**

**The type: WORD1The word :gyuunyuu**

**==========Lexical Error: milk is not valid token=====================**

**The type: ERRORThe word :milk**

**The type: WORD1The word :sukiyaki**

**The type: WORD1The word :tenpura**

**The type: WORD1The word :sushi**

**The type: WORD1The word :biiru**

**==========Lexical Error: beer is not valid token=====================**

**The type: ERRORThe word :beer**

**The type: WORD1The word :sake**

**The type: WORD1The word :tokyo**

**The type: WORD1The word :kyuushuu**

**==========Lexical Error: Osaka is not valid token=====================**

**The type: ERRORThe word :Osaka**

**The type: WORD1The word :choucho**

**==========Lexical Error: butterfly is not valid token=====================**

**The type: ERRORThe word :butterfly**

**The type: WORD1The word :an**

**The type: WORD1The word :idea**

**The type: WORD1The word :yasashii**

**==========Lexical Error: easy is not valid token=====================**

**The type: ERRORThe word :easy**

**The type: WORD1The word :muzukashii**

**==========Lexical Error: difficult is not valid token=====================**

**The type: ERRORThe word :difficult**

**The type: WORD1The word :ureshii**

**==========Lexical Error: pleased is not valid token=====================**

**The type: ERRORThe word :pleased**

**The type: WORD1The word :shiawase**

**==========Lexical Error: happy is not valid token=====================**

**The type: ERRORThe word :happy**

**The type: WORD1The word :kanashii**

**==========Lexical Error: sad is not valid token=====================**

**The type: ERRORThe word :sad**

**The type: WORD1The word :omoi**

**==========Lexical Error: heavy is not valid token=====================**

**The type: ERRORThe word :heavy**

**The type: WORD1The word :oishii**

**==========Lexical Error: delicious is not valid token=====================**

**The type: ERRORThe word :delicious**

**The type: WORD1The word :tennen**

**==========Lexical Error: natural is not valid token=====================**

**The type: ERRORThe word :natural**

**The type: WORD2The word :nakI**

**==========Lexical Error: cry is not valid token=====================**

**The type: ERRORThe word :cry**

**The type: WORD2The word :ikI**

**==========Lexical Error: go\* is not valid token=====================**

**The type: ERRORThe word :go\***

**The type: WORD2The word :tabE**

**==========Lexical Error: eat is not valid token=====================**

**The type: ERRORThe word :eat**

**The type: WORD2The word :ukE**

**==========Lexical Error: take\* is not valid token=====================**

**The type: ERRORThe word :take\***

**The type: WORD2The word :kakI**

**==========Lexical Error: write is not valid token=====================**

**The type: ERRORThe word :write**

**The type: WORD2The word :yomI**

**==========Lexical Error: read is not valid token=====================**

**The type: ERRORThe word :read**

**The type: WORD2The word :nomI**

**==========Lexical Error: drink is not valid token=====================**

**The type: ERRORThe word :drink**

**The type: WORD2The word :agE**

**==========Lexical Error: give is not valid token=====================**

**The type: ERRORThe word :give**

**The type: WORD2The word :moraI**

**==========Lexical Error: receive is not valid token=====================**

**The type: ERRORThe word :receive**

**The type: WORD2The word :butsI**

**==========Lexical Error: hit is not valid token=====================**

**The type: ERRORThe word :hit**

**The type: WORD2The word :kerI**

**==========Lexical Error: kick is not valid token=====================**

**The type: ERRORThe word :kick**

**The type: WORD2The word :shaberI**

**==========Lexical Error: talk is not valid token=====================**

**The type: ERRORThe word :talk**

**The type: EOFMThe word :eofm**

**Section 4: Factored rules with new non-terminal names**

****

**Section 5: Parser code**

****

****

****

****

****

****

****

****

****

****

****

****

****

****

****

**Section 6: Parser test results**

**Test1:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: N**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is watashi**

**PROCESSING NOUN**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is rika**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is desu**

**PROCESSING BE**

**Matched IS**

**DESCRIPTION: rika**

**TENSE: IS**

**The current word is .**

**Matched PERIOD**

**The current word is watashi**

**======================PROCESSING BEGINNING STORY=====================**

**PROCESSING NOUN**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is sensei**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is desu**

**PROCESSING BE**

**Matched IS**

**DESCRIPTION: sensei**

**TENSE: IS**

**The current word is .**

**Matched PERIOD**

**The current word is rika**

**======================PROCESSING BEGINNING STORY=====================**

**PROCESSING NOUN**

**Matched WORD1**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: rika**

**PROCESSING AFTER SUBJECT**

**The current word is gohan**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is o**

**Matched OBJECT**

**OBJECT: gohan**

**The current word is tabE**

**PROCESSING VERB**

**Matched WORD2**

**ACTION: tabE**

**PROCESSING TENSE**

**The current word is masu**

**Matched VERB**

**TENSE: VERB**

**The current word is .**

**Matched PERIOD**

**The current word is watashi**

**======================PROCESSING BEGINNING STORY=====================**

**PROCESSING NOUN**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is tesuto**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is o**

**Matched OBJECT**

**OBJECT: tesuto**

**The current word is seito**

**PROCESSING NOUN**

**Matched WORD1**

**The current word is ni**

**Matched DESTINATION**

**TO: seito**

**PROCESSING VERB**

**The current word is agE**

**Matched WORD2**

**ACTION: agE**

**PROCESSING TENSE**

**The current word is mashita**

**Matched VERBPAST**

**TENSE: VERBPAST**

**The current word is .**

**Matched PERIOD**

**The current word is shikashi**

**======================PROCESSING BEGINNING STORY=====================**

**Matched CONNECTOR**

**CONNECTOR: However**

**PROCESSING NOUN**

**The current word is seito**

**Matched WORD1**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: seito**

**PROCESSING AFTER SUBJECT**

**The current word is yorokobI**

**PROCESSING VERB**

**Matched WORD2**

**ACTION: yorokobI**

**PROCESSING TENSE**

**The current word is masendeshita**

**Matched VERBPASTNEG**

**TENSE: VERBPASTNEG**

**The current word is .**

**Matched PERIOD**

**The current word is dakara**

**======================PROCESSING BEGINNING STORY=====================**

**Matched CONNECTOR**

**CONNECTOR: Therefore**

**PROCESSING NOUN**

**The current word is watashi**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is kanashii**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is deshita**

**PROCESSING BE**

**Matched WAS**

**DESCRIPTION: kanashii**

**TENSE: WAS**

**The current word is .**

**Matched PERIOD**

**The current word is soshite**

**======================PROCESSING BEGINNING STORY=====================**

**Matched CONNECTOR**

**CONNECTOR: Then**

**PROCESSING NOUN**

**The current word is rika**

**Matched WORD1**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: rika**

**PROCESSING AFTER SUBJECT**

**The current word is toire**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is ni**

**Matched DESTINATION**

**TO: toire**

**PROCESSING VERB**

**The current word is ikI**

**Matched WORD2**

**ACTION: ikI**

**PROCESSING TENSE**

**The current word is mashita**

**Matched VERBPAST**

**TENSE: VERBPAST**

**The current word is .**

**Matched PERIOD**

**The current word is rika**

**======================PROCESSING BEGINNING STORY=====================**

**PROCESSING NOUN**

**Matched WORD1**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: rika**

**PROCESSING AFTER SUBJECT**

**The current word is nakI**

**PROCESSING VERB**

**Matched WORD2**

**ACTION: nakI**

**PROCESSING TENSE**

**The current word is mashita**

**Matched VERBPAST**

**TENSE: VERBPAST**

**The current word is .**

**Matched PERIOD**

**The current word is eofm**

**=====END-Prasering=====**

**Test2:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: n**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is soshite**

**Matched CONNECTOR**

**CONNECTOR: Then**

**PROCESSING NOUN**

**The current word is watashi**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is rika**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is desu**

**PROCESSING BE**

**Matched IS**

**DESCRIPTION: rika**

**TENSE: IS**

**The current word is ne**

**SYNTAX ERROR: EXPECTING PERIOD BUT FOUND ne**

**Test3:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: n**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is dakara**

**Matched CONNECTOR**

**CONNECTOR: Therefore**

**PROCESSING NOUN**

**The current word is watashi**

**Matched PRONOUN**

**The current word is de**

**SYNTAX ERROR: EXPECTING SUBJECT BUT FOUND de**

**Test4:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: n**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is watashi**

**PROCESSING NOUN**

**Matched PRONOUN**

**The current word is wa**

**Matched SUBJECT**

**ACTOR: I/me**

**PROCESSING AFTER SUBJECT**

**The current word is rika**

**PROCESSING NOUN**

**Matched WORD1**

**PROCESSING AFTER NOUN**

**The current word is mashita**

**SYNTAX ERROR: UNEXPECTED WORD: mashita FOUND IN afterNOUN5:**

**Test5:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: n**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is wa**

**PROCESSING NOUN**

**wa is not connector or noun -none of the alternative fit**

**SYNTAX ERROR: UNEXPECTED WORD: wa FOUND IN noun**

**Test 6:**

**Parser Funciton**

**Enter Filename of the File you want to parse:**

**test.txt**

**KEEP TRACK OF PARSING PROCESS? Y / N: n**

**===============PROCESSING STORY======================**

**======================PROCESSING BEGINNING STORY=====================**

**The current word is apple**

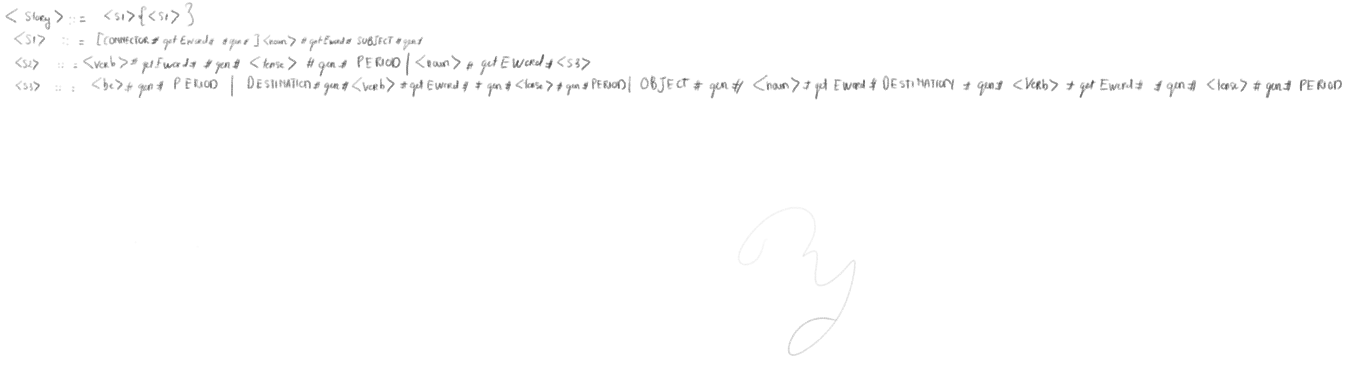
**Lexical Error: apple is not valid token**

**PROCESSING NOUN**

**apple is not connector or noun -none of the alternative fit**

**SYNTAX ERROR: UNEXPECTED WORD: apple FOUND IN noun**

**Section 7: Updated Parser code for translation**

****

**Section 7: Semantic functions commented with the functionality and the name of the author.**

****



**Section 8: Semantic test results**

Translated.txt

ACTOR: I/me

DESCRIPTION: rika

TENSE: IS

ACTOR: I/me

DESCRIPTION: sensei

TENSE: IS

ACTOR: rika

OBJECT: gohan

ACTION: tabE

TENSE: VERB

ACTOR: I/me

OBJECT: tesuto

TO: seito

ACTION: agE

TENSE: VERBPAST

CONNECTOR: However

ACTOR: seito

ACTION: yorokobI

TENSE: VERBPASTNEG

CONNECTOR: Therefore

ACTOR: I/me

DESCRIPTION: kanashii

TENSE: WAS

CONNECTOR: Then

ACTOR: rika

TO: toire

ACTION: ikI

TENSE: VERBPAST

ACTOR: rika

ACTION: nakI

TENSE: VERBPAST