Andre Ibrahim
40132881

SECTION 1: Attribute grammar

START -> SEMANTICEPSILON REPTSTART0 SEMANTICCLASSDECLORFUNCDEF .
SEMANTICCLASSDECLORFUNCDEF -> .

APARAMS -> EXPR SEMANTICEPSILON REPTAPARAMS1 SEMANTICAPARAMS .
APARAMS -> .
SEMANTICAPARAMS -> .

APARAMSTAIL -> comma EXPR .

ADDOP -> plus .
ADDOP -> minus .
ADDOP -> or .

ARITHEXPR -> TERM SEMANTICEPSILON RIGHTRECARITHEXPR SEMANTICARITHEXPR
.
SEMANTICARITHEXPR -> .

ARRAYSIZE -> lsqbr ARRAYSIZE2 .
ARRAYSIZE2 -> intlit SEMANTICINTLIT rsqbr .
ARRAYSIZE2 -> rsqbr .
SEMANTICINTLIT -> .

ASSIGNOP -> equal .

CLASSDECL -> SEMANTICEPSILON class id SEMANTICTOKEN OPTCLASSDECL2 lcurbr
REPTCLASSDECL4 SEMANTICCLASSDECL rcurbr semi .
SEMANTICCLASSDECL -> .

CLASSDECLORFUNCDEF -> CLASSDECL .
CLASSDECLORFUNCDEF -> FUNCDEF .

EXPR -> ARITHEXPR EXPR2 SEMANTICEXPR .

EXPR2 -> RELOP SEMANTICTOKEN ARITHEXPR .
EXPR2 -> .
SEMANTICEXPR -> .

FPARAMS -> SEMANTICEPSILON id SEMANTICTOKEN colon TYPE SEMANTICTOKEN
SEMANTICEPSILON REPTFPARAMS3 SEMANTICARRAYSIZE REPTFPARAMS4
SEMANTICFPARAMS .

FPARAMS -> .
SEMANTICFPARAMS -> .

FPARAMSTAIL -> comma id SEMANTICTOKEN colon TYPE SEMANTICTOKEN
SEMANTICEPSILON REPTFPARAMS3 SEMANTICARRAYSIZE .

FACTOR -> SEMANTICEPSILON FUNCTIONCALLORVARIABLE
SEMANTICFACTORCALLORCAR .
FACTOR -> intlit SEMANTICTOKEN SEMANTICFACTOR .
FACTOR -> floatlit SEMANTICTOKEN SEMANTICFACTOR .
FACTOR -> lpar ARITHEXPR rpar SEMANTICFACTOR .
FACTOR -> not SEMANTICTOKEN FACTOR SEMANTICFACTOR .
FACTOR -> SIGN SEMANTICTOKEN FACTOR SEMANTICFACTOR .
SEMANTICTOKEN -> .
SEMANTICFACTOR -> .
SEMANTICFACTORCALLORCAR -> .

FUNCBODY -> lcurbr SEMANTICEPSILON REPTFUNCBODY1 SEMANTICFUNCBODY rcurbr
.
SEMANTICFUNCBODY -> .

FUNCDEF -> SEMANTICEPSILON FUNCHEAD FUNCBODY SEMANTICFUNCDEF .
SEMANTICFUNCDEF -> .

FUNCHEAD -> function id SEMANTICTOKEN FUNCHEAD3 .

FUNCHEAD2 -> id SEMANTICTOKEN lpar FPARAMS rpar arrow RETURNTYPE
SEMANTICTOKEN SEMANTICFUNCARROW .
FUNCHEAD2 -> constructor lpar FPARAMS rpar SEMANTICFUNCCONSTSTRUCT .
SEMANTICFUNCARROW -> .
SEMANTICFUNCCONSTSTRUCT -> .

FUNCHEAD3 -> SEMANTICEPSILON sr FUNCHEAD2 .
FUNCHEAD3 -> SEMANTICEPSILON lpar FPARAMS rpar arrow RETURNTYPE
SEMANTICTOKEN SEMANTICFUNCARROW .

VARIABLE -> id SEMANTICTOKEN SEMANTICEPSILON VARIABLE3 SEMANTICVARIABLE .
VARIABLE3 -> INDICE .
VARIABLE3 -> VARIABLE2 .
VARIABLE3 -> .
VARIABLE2 -> dot SEMANTICTOKEN id SEMANTICEPSILON VARIABLE4
SEMANTICVARIABLE .
VARIABLE4 -> lpar APARAMS rpar VARIABLE2 .
VARIABLE4 -> INDICE VARIABLE5 .

VARIABLE5 -> VARIABLE2 .
VARIABLE5 -> .

FUNCTIONCALLORVARIABLE -> id SEMANTICTOKEN FUNCTIONCALLORVARIABLE1 .
FUNCTIONCALLORVARIABLE1 -> SEMANTICEPSILON SEMANTICEPSILON INDICELOOP
SEMANTICINDICELIST SEMANTICVARIABLE FUNCTIONCALLORVARIABLE2 .
FUNCTIONCALLORVARIABLE1 -> SEMANTICEPSILON lpar APARAMS rpar
SEMANTICFUNCTIONCALL FUNCTIONCALLORVARIABLE2 .
FUNCTIONCALLORVARIABLE2 -> dot id SEMANTICTOKEN FUNCTIONCALLORVARIABLE3
.
FUNCTIONCALLORVARIABLE2 -> .
FUNCTIONCALLORVARIABLE3 -> SEMANTICEPSILON SEMANTICEPSILON INDICELOOP
SEMANTICINDICELIST SEMANTICVARIABLE FUNCTIONCALLORVARIABLE2 .
FUNCTIONCALLORVARIABLE3 -> SEMANTICEPSILON lpar APARAMS rpar
SEMANTICFUNCTIONCALL FUNCTIONCALLORVARIABLE2 .
SEMANTICVARIABLE -> .
SEMANTICFUNCTIONCALL -> .

INDICE -> lsqbr ARITHEXPR rsqbr .

LOCALVARDECL -> localvar id SEMANTICID colon TYPE SEMANTICTYPE
LOCALVARDECL2 .
LOCALVARDECL2 -> SEMANTICEPSILON REPTFPARAMS3 SEMANTICARRAYSIZE
SEMANTICVARDECL semi .
LOCALVARDECL2 -> lpar APARAMS rpar SEMANTICVARDECL semi .
SEMANTICID -> .
SEMANTICTYPE -> .
SEMANTICEPSILON -> .
SEMANTICARRAYSIZE -> .
SEMANTICVARDECL -> .

LOCALVARDECLORSTMT -> LOCALVARDECL .
LOCALVARDECLORSTMT -> STATEMENT .

MEMBERDECL -> SEMANTICEPSILON MEMBERFUNCDECL .
MEMBERDECL -> SEMANTICEPSILON MEMBERVARDECL .

MEMBERFUNCDECL -> function id SEMANTICTOKEN colon lpar FPARAMS rpar arrow
RETURNTYPE SEMANTICTOKEN SEMANTICMEMBERFUNCDECL semi .
MEMBERFUNCDECL -> constructor colon lpar FPARAMS rpar
SEMANTICMEMBERFUNCDECL semi .
SEMANTICMEMBERFUNCDECL -> .

MEMBERVARDECL -> attribute id SEMANTICTOKEN colon TYPE SEMANTICTOKEN
SEMANTICEPSILON REPTFPARAMS3 SEMANTICARRAYSIZE
SEMANTICMEMBERVARDECL semi .
SEMANTICMEMBERVARDECL -> .

MULTOP -> mult .
MULTOP -> div .
MULTOP -> and .

OPTCLASSDECL2 -> isa SEMANTICEPSILON id SEMANTICTOKEN
REPTOPTCLASSDECL22 SEMANTICISA .
OPTCLASSDECL2 -> .
SEMANTICISA -> .

RELEXPR -> ARITHEXPR RELOP SEMANTICTOKEN ARITHEXPR SEMANTICRELEXPR .
SEMANTICRELEXPR -> .

RELOP -> eq .
RELOP -> neq .
RELOP -> lt .
RELOP -> gt .
RELOP -> leq .
RELOP -> geq .

REPTSTART0 -> CLASSDECLORFUNCDEF REPTSTART0 .
REPTSTART0 -> .

REPTAPARAMS1 -> APARAMSTAIL REPTAPARAMS1 .
REPTAPARAMS1 -> .

REPTCLASSDECL4 -> VISIBILITY SEMANTICTOKEN MEMBERDECL REPTCLASSDECL4 .
REPTCLASSDECL4 -> .

REPTFPARAMS3 -> ARRAYSIZE REPTFPARAMS3 .
REPTFPARAMS3 -> .

REPTFPARAMS4 -> FPARAMSTAIL REPTFPARAMS4 .
REPTFPARAMS4 -> .

REPTFUNCBODY1 -> LOCALVARDECLORSTMT REPTFUNCBODY1 .
REPTFUNCBODY1 -> .

REPTOPTCLASSDECL22 -> comma id SEMANTICTOKEN REPTOPTCLASSDECL22 .
REPTOPTCLASSDECL22 -> .

REPTSTATBLOCK1 -> STATEMENT REPTSTATBLOCK1 .
REPTSTATBLOCK1 -> .

RETURNTYPE -> TYPE .
RETURNTYPE -> void .

RIGHTRECARITHEXPR -> .
RIGHTRECARITHEXPR -> ADDOP SEMANTICTOKEN TERM RIGHTRECARITHEXPR .

RIGHTRECTERM -> .
RIGHTRECTERM -> MULTOP SEMANTICTOKEN FACTOR RIGHTRECTERM .

SIGN -> plus .
SIGN -> minus .

STATBLOCK -> lcurbr SEMANTICEPSILON REPTSTATBLOCK1 rcurbr
SEMANTICSTATBLOCK .
STATBLOCK -> STATEMENT .
STATBLOCK -> .
SEMANTICSTATBLOCK -> .

STATEMENT -> FUNCTIONCALLORASIGNSTAT semi .
STATEMENT -> SEMANTICEPSILON if lpar RELEXPR rpar then STATBLOCK else
STATBLOCK SEMANTICIFSTAT semi .
STATEMENT -> SEMANTICEPSILON while lpar RELEXPR rpar STATBLOCK
SEMANTICWHILESTAT semi .
STATEMENT -> read lpar VARIABLE rpar SEMANTICREADSTAT semi .
STATEMENT -> write lpar EXPR rpar SEMANTICWRITESTAT semi .
STATEMENT -> return lpar EXPR rpar SEMANTICRETURNSTAT semi .
SEMANTICRETURNSTAT -> .
SEMANTICWRITESTAT -> .
SEMANTICREADSTAT -> .
SEMANTICIFSTAT -> .
SEMANTICWHILESTAT -> .

FUNCTIONCALLORASIGNSTAT -> SEMANTICEPSILON id SEMANTICTOKEN
ISFUNCTIONCALLORVARIABLE .

ISFUNCTIONCALLORVARIABLE -> lpar APARAMS rpar AFTERFUNCTIONCALL .
ISFUNCTIONCALLORVARIABLE -> SEMANTICEPSILON INDICELOOP
SEMANTICINDICELIST AFTERVARIABLE .

AFTERFUNCTIONCALL -> dot id SEMANTICTOKEN MIDDLESTATE .

AFTERVARIABLE -> dot id SEMANTICTOKEN MIDDLESTATE .

MIDDLESTATE -> SEMANTICEPSILON INDICELOOP SEMANTICINDICELIST
AFTERVARIABLE .
MIDDLESTATE -> lpar APARAMS rpar AFTERFUNCTIONCALL .

AFTERVARIABLE -> ENDASSIGN .
AFTERFUNCTIONCALL -> SEMANTICFUNCTIONCALLSTAT.

INDICELOOP -> INDICE INDICELOOP .
INDICELOOP -> .

ENDASSIGN -> ASSIGNOP SEMANTICTOKEN EXPR SEMANTICASSIGNSTAT.
SEMANTICINDICELIST -> .
SEMANTICASSIGNSTAT -> .
SEMANTICFUNCTIONCALLSTAT -> .

TERM -> SEMANTICEPSILON FACTOR RIGHTRECTERM SEMANTICTERM .
SEMANTICTERM -> .

TYPE -> integer .
TYPE -> float .
TYPE -> id .

VISIBILITY -> public .
VISIBILITY -> private .
VISIBILITY -> .

List of semantic actions

SEMANTICEPSILON
This is pushed on to the semantic stack to allow a pop until epsilon operation
SEMANTICTOKEN
This action creates a leaf of the current token only keeps the relevant tokens for example id or intlit for the purpose of having the abstract syntax tree
SEMANTICVARDECL
Pop 3 times from the semantic stack and create subtree then push the created subtree
SEMANTICARRAYSIZE
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICRETURNSTAT
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICEXPR
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICARITHEXPR

Pop until epsilon then
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICTERM
Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICFACTOR
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICVARIABLE
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICFACTORCALLORCAR
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICFUNCTIONCALL
Pop until epsilon then
Pop 1 time from the semantic stack and create subtree then push the created subtree

SEMANTICAPARAMS
Pop until epsilon then
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICWRITESTAT
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICREADSTAT
Pop 1 time from the semantic stack and create subtree then push the created subtree
SEMANTICSTATBLOCK
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICFUNCBODY
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICINDICELIST
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICASSIGNSTAT
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICFUNCTIONCALLSTAT
Pop until epsilon from the semantic stack and create subtree then push the created subtree
SEMANTICRELEXPR
Pop 3 time from the semantic stack and create subtree then push the created subtree

SEMANTICIFSTAT

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICWHILESTAT

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICFUNCDEF

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICFUNCARROW

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICFUNCCONSTSTRUCT

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICFPARAMS

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICCLASSDECL

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICMEMBERFUNCDECL

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICMEMBERVARDECL

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICISA

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SEMANTICCLASSDECLORFUNCDEF

Pop until epsilon from the semantic stack and create subtree then push the created subtree

SECTION 2: Design

For my solution I updated the parser and created two new classes and an Enum: a Tree class, a TreeFactory class and a Concept enum .

First of all we needed to use a tree data structure for the AST, it is basically a class that contains a list of its own type. I added a couple methods to be able to peak into the tree via print statement so that we can see what the structure looks like.

The Concept enum holds the list of semantic actions since they are considered concepts within our grammar, this enum is meant to decouple the code from the grammar even though right now it matches one to one it doesn't necessarily need to in the future.

In the parser, inside of our parsing loop before we analyze the top we check if it's a semantic action, if it is then we need to run the actions defined above.

To achieve the semantic action I created a TreeFactory which is the design pattern factoryMethod and allows me to build a subtree based on the current semantic action and the current content of the semantic stack.

SECTION 3: Use of Tools

Tools used in grammar transformation:

1. To get the parsing table used the university of calgary tool https://smlweb.cpsc.ucalgary.ca/start.html For some context I needed to do that since inorder to make it easier to inject the semantic actions I added them to my grammar as nullable nonTerminals.

2. The ucal tool generates the parsing table as html and we can get that table and put it through this tool https://www.convertcsv.com/html-table-to-csv.htm to convert the tables into csv format.

3. Finally at some point I had an issue with the ucal tool telling me that the url request was too short to parse my grammar so I used this tool to condense it before feeding it to ucal https://www.removelinebreaks.net/

Tools used in code:

1. The only tool used in the code is the Lexer that I had built in assignment 1 everything else is vanilla Typescript.