**CS 419 Compiler**

**Project Form**

**Project Idea:**

……project 3……………………………….

**Team Members NO#:** …………………………………..

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Level& Department** | **Section(Dayfrom-to)** | **Role (Lead/Member)** | **Grade** |
| 201900959 | ياسمين عادل نصر غانم | 3 cs |  |  |  |
| 201900012 | ابراهيم محمود مكي حسين | 3 cs |  |  |  |
| 201900437 | عبدالرحمن محمد فكري ثابت | 3 cs |  |  |  |
| 20180142 | إيثار طارق عبد العزيز | 4 cs |  |  |  |
| 201900984 | يوسف شعبان إمام | 3 cs |  |  |  |
| 20180689 | ياسمين محمد حسن | 4 cs |  |  |  |
| 201900496 | عماد علي عماد حافظ | 3 cs |  |  |  |

**Scanning part :**

**Regular Expression, Finite automata and Conversion from RegX to NFA, NFA to DFA**

A piece of paper with writing on it

Description automatically generated Diagram

Description automatically generated

A drawing on a piece of paper

Description automatically generated with medium confidence

Diagram

Description automatically generated A picture containing diagram

Description automatically generated

A picture containing text, electronics

Description automatically generatedGraphical user interface, application

Description automatically generated

Diagram

Description automatically generatedDiagram, schematic

Description automatically generated Diagram

Description automatically generated

Diagram

Description automatically generatedDiagram

Description automatically generated

Diagram, schematic

Description automatically generated Diagram, schematic

Description automatically generated

Diagram, schematic

Description automatically generatedDiagram

Description automatically generated

**Transition\_Table**

D1,d2,d8.d11,d12,d13,d14,d15,d16,d17,d18,d19,d36,d30,d21,d37,d25,d27,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | letter | digit | @ / ^ | $ / # | < | > | / | = | ! | ~ | { | } | [ | ] | - | + | \* | , | “ | | | & | ; | New line | Tap / space |
| D0 | D3 | D4 | D1 | D2 | D7 | D23 | D18 | D24 | D20 | D16 | D17 | D19 | D15 | D30 | D12 | D13 | D31 | D14 | D36 | D26 | D28 | AC | AC | AC |
| D1 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D2 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D3 | D3 | D3 | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D4 | error | D5 | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D5 | error | D5 | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D7 | error | error | error | error | error | error | D9 | D8 | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D8 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D9 | D9 | D9 | D9 | D9 | D9 | D9 | D10 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 |
| D10 | D9 | D9 | D9 | D9 | D9 | D11 | D10 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 |
| D11 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D12 | error | error | error | error | error | D38 | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D13 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D14 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D15 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D16 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D17 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D18 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D19 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D36 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D30 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D20 | error | error | error | error | error | error | error | D21 | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D21 | error | error | error | error | error | error | error | error | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D23 | error | error | error | error | error | error | error | D37 | error | Error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D37 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D24 | error | error | error | error | error | error | error | D25 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D25 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D26 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | D27 | error | error | error | error |
| D27 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | AC | AC |
| D28 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | D28 | error | AC | AC |
| D31 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | D32 | error | error | error | error | error | AC | AC |
| D32 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | D33 | error | error | error |
| D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | AC | D33 |
| D38 | D3 | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error | error |

@ / ^ / $ / # / ~ / { / } / [ / ] /+ / ‘ /” = symbol (d1) =

d 44 error

AC39 symbol (d1) =

Ac 40 identifer

Ac41 digit

AC42 comment

Ac 43 error

D1,d2,d8., ,d21,d37,d25,d27,d29

D1, D2, d12,d13,d14,d15,d16,d17,d18,d19,d36,d30

D23,d24

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | letter | digit | symbol | < | > | / | = | ! | - | \* | | | & | ; | New line | Tap / space |
| D0 | D3 | D4 | D1 | D7 | D23 | D18 | D23 | D20 | D12 | D31 | D26 | D28 | AC | AC | AC |
| D1.d2 | error | error | error | error | error | error | error | error | error | error | error | error | error | AC39 | AC39 |
| D3 | D3 | D3 | error | error | error | error | error | error | error | error | error | error | error | AC40 | AC40 |
| D4 | error | D5 | error | error | error | error | error | error | error | error | error | error | error | AC41 | AC41 |
| D5 | error | D5 | error | error | error | error | error | error | error | error | error | error | error | AC41 | AC41 |
| D7 | error | error | error | error | error | D9 | D8 | error | error | error | error | error | error | AC39 | AC39 |
| D9 | D9 | D9 | D9 | D9 | D9 | D10 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 |
| D10 | D9 | D9 | D9 | D9 | D11 | D10 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 |
| D11 | error | error | error | error | error | error | error | error | error | error | error | error | error | AC42 | AC42 |
| D20 | error | error | error | error | error | error | D1 | error | error | error | error | error | error | error | error |
| D23 | error | error | error | error | error | error | D1 | error | error | error | error | error | error | AC39 | AC39 |
| D26 | error | error | error | error | error | error | error | error | error | error | D1 | error | error | error | error |
| D28 | error | error | error | error | error | error | error | error | error | error | error | D1 | error | error | error |
| D31 | error | error | error | error | error | error | error | error | error | D32 | error | error | error | AC39 | AC39 |
| D32 | error | error | error | error | error | error | error | error | error | D33 | error | D33 | error | error | error |
| D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | D33 | AC42 | D33 |
| D38 | D3 | error | error | error | error | error | error | error | error | error | error | error | error | error | error |
| D error | error | error | error | error | error | error | error | error | error | error | error | error | error | Ac error | Ac error |

**Parsing Part :**

**Note you can see the parse table in exill file in git hub**

**Parser grammar rule :**

1. Program →Start-Symbols ClassDeclaration End-Symbols.

2. Start-Symbols →@| ^

3. End-Symbols→$ |#

4. ClassDeclaration→ Type ID{ Class\_Implementation} | Type ID Infer {

Class\_Implementation}

5. Class\_Implementation→ Variable\_Decl Class\_Implementation|

Method\_Decl Class\_Implementation | Comment Class\_Implementation |

require\_command Class\_Implementation| Func \_Call

Class\_Implementation |em

6. Method\_Decl→ Func Decl ;| Func Decl { Variable\_Decl Statements }

7. Func Decl →Type ID (ParameterList)

8. Type → Ipok |Sipok |Craf |Sequence |Ipokf |Sipokf |Valueless |Rational

9. ParameterList →em| None | Non-Empty List

10. Non-Empty List→ Type ID Non-Empty List \*

Non-Empty List\* → Type ID Non-Empty List\* | e

11. Variable\_Decl→ em | Type ID\_List ; Variable\_Decl | Type ID\_List [ID] ; Variable\_Decl

12. ID\_List →ID ID\_List \*

ID\_List\* → ID ID\_List \* | e

13. Statements→em | Statement Statements

14. Statement→Assignment | If \_Statement | However \_Statement |

when\_Statement | Respondwith \_ Statement | Endthis

\_Statement|Scanvalur (ID ); | Print (Expression); |

15. Assignment→ Variable\_Decl = Expression;

16. Func \_Call → ID (Argument\_List) ;

17. Argument\_List →em | NonEmpty\_Argument\_List

18. NonEmpty\_Argument\_List →Expression NonEmpty\_Argument\_List \*

NonEmpty\_Argument\_List ,\* → Expression NonEmpty\_Argument\_List \* | e

19. Block Statements→{ statements }

20. If \_Statement→ if (Condition \_Expression) Block Statements | if

(Condition \_Expression) Block Statements else Block Statements

21. Condition \_Expression→ Condition |Condition Condition \_Op Condition

22. Condition \_Op → && | ||

23. Condition→ Expression Comparison \_Op Expression

24. Comparison \_Op → == | != | > | >= | < | <=

25. However \_Statement → However (Condition \_Expression) Block Statements

26. when \_Statement → when ( expression ; expression ; expression ) Block Statements

27. Respondwith \_Statement→ Respondwith Expression ; | return ID ;

28. Endthis \_Statement→ Endthis;

29. Expression → Term Expression \*

Expression\* → Add\_Op Term Expression\* | e

30. Add\_Op → + | -

31. Term→Factor Term \*

Term \* → Mul\_Op Factor Term \* |e

32. Mul\_Op→\* | /

33. Factor→ ID| Number

34. Comment →</ STR /> | \*\*\*STR

35. Require\_command →Require(F\_name.txt);

36. F\_name →STR

**First & follow part**

First calculation :

First(Program) → @, ^

First(Start-Symbols) → @, ^

First(End-Symbols)→ $ ,#

First(ClassDeclaration)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

First(Class\_Implementation)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require , e

First(Method\_Decl)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

First( Func Decl)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

First(Type)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

First(ParameterList)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , e ,None

First(Non-Empty List)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , e

First(Non-Empty List\*)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , e

First(Variable\_Decl)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , e

First(ID\_List)→ ID

First(ID\_List \*)→ e

First(Statements)→ e , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when\_Statement , Respondwith, Endthis , Scanvalur, Print,

First(Statement)→ e , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If , However ,when, Respondwith , Endthis , Scanvalur, Print,

First(Assignment)→ e , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

First(Func \_Call)→ ID

First(Argument\_List)→ em , ID, Number

First(NonEmpty\_Argument\_List)→ ID, Number

First(NonEmpty\_Argument\_List\*)→ ID, Number ,e

First(Block Statements)→ {

First(If \_Statement)→ if

First(Condition \_Expression)→ ID , Number

First(Condition \_Op)→ && , ||

First(Condition)→ ID , Number

First(Comparison \_Op)→ == , != , > , >= , < , <=

First(However \_Statement)→ However

First(when \_Statement)→ when

First(Respondwith \_Statement)→ Respondwith , return

First(Endthis)→ Endthis

First(Expression)→ ID , Number

First(Expression\*)→ + , - , e

First(Term)→ ID , Number

First(Expression\*)→ \* , / ,e

First(Factor)→ ID , Number

First(Comment) →</ STR /> , \*\*\*STR

First(Require\_command) → Require

First(F\_name) → STR

Follow calculation :

Follow (Program ) → $

Follow (Start-Symbols) → First(ClassDeclaration)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational

Follow (End-Symbols) → $

Follow (ClassDeclaration) → {

Follow ( Class\_Implementation) → First(Class\_Implementation)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require

Follow (Method\_Decl) → ; , {

Follow (Func Decl) → (

Follow (type) → ID

Follow (ParameterList) → )

Follow (Non-Empty List) → )

Follow (Non-Empty List\*) → )

Follow (. Variable\_Decl) → { , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require = ,If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (. ID\_List) → ; , [

Follow (. ID\_List\*) → ; , [

Follow (. Statements) → } ,

Follow (. Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (Assignment) → } Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (Func \_Call) → First(Class\_Implementation)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require ,

Follow (Argument\_List) → )

Follow (NonEmpty\_Argument\_List) → )

Follow (NonEmpty\_Argument\_List\*) → )

Follow (Block Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print, else ,

Follow (IF\_statment) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when\_ , Respondwith, Endthis , Scanvalur, Print,

Follow (Condition \_Expression) → )

Follow (Condition \_Op) → ID , Number

Follow (Condition) → && , || , )

Follow (Comparison \_Op) → ID , Number

Follow (However\_ Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (When\_ Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (Respondwith \_Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when, Respondwith, Endthis , Scanvalur, Print,

Follow (Endthis \_Statement) → } , Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , If, However ,when\_ , Respondwith, Endthis , Scanvalur, Print,

Follow (Expression) → } ; ) == , != , > , >= , < , <= ID , Number

Follow (Expression\*) → } ; ) == , != , > , >= , < , <= ID , Number

Follow (Add\_Op) → ID , Number

Follow (Term) → + , -

Follow (Term\*) → + , -

Follow (Mul\_Op) → ID , Number

Follow (comment)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require ,

Follow (. Require\_command)→ Ipok ,Sipok ,Craf ,Sequence ,Ipokf ,Sipokf ,Valueless ,Rational , ID ,</ STR /> , \*\*\*STR , Require ,

Follow ( F\_name)→ .