L'École Nationale Supérieure d'Informatique
et d'Analyse des Systèmes

Projet compilation

Réalisation d'un compilateur du langage R avec le langage C

Filière Génie Logiciel

Réaliser par :
Abdel Moumene HADFI
Anas KHALID
Hamza BOUQDIR
Ahmed HAFDI

Encadrer par :
M. Younes TABII
M. Rachid OULAD HAJ THAMI

13-02-2020

Remerciements

C'est avec un grand plaisir que nous réservons ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui, de près ou de loin, ont contribué à la réalisation et l'aboutissement de ce travail.

On saisit l'occasion pour remercier tous nos enseignants à l'ENSIAS qu'on offert des formations très complètes ce qui nous a aidé a bien aboutir notre projet, et bien précisément notre cours Compilation. Nous tenons également à exprimer notre sincère gratitude envers tous ceux qui nous ont aidé ou ont participé au bon déroulement de ce projet.

Enfin, nous exprimons nos plus vifs remerciements à notre enseignants et encadrants **M.Younes TABII** et **M.Rachid OULAD HAJ THAMI** , pour leurs précieux conseils et leurs encourage- ments au cours de la préparation de ce projet.

Table des figures

Table des matières

1. Introduction

Afin d'appliquer les méthodologies et les notions enseignées durant le cours compilation, nous sommes invités à réaliser un projet qui nous permet d'appliquer nos connaissances théoriques sur le champ pratique.

il s'agit d'un compilateur de langage R avec le langage c. Le présent rapport est dédié à la présentation de l'ensemble des travaux menés dans le cadre de notre projet. La première partie est destinée à la présentation du grammaire LL(1) et au lexique. La deuxième partie aborde la réalisation de l'analyseur lexical. La troisième partie est destinée à la réalisation de l'analyseur lexical. La dernière partie est destinée à décrire les règles sémantique. Enfin, la conclusion générale résume le bilan du travail effectué et les principales perspectives.

2. lexique et grammaire

2.1 Grammaire LL(1)

1. ***INST*** *: ID B| BOUCLE | PRINT | SI*
2. ***AFTER_ID*** *: AFF_LEFT | (EXPR | epsilon) C(AFF_RIGHT,EXPR)*
3. ***C(AFF_RIGHT,EXPR)*** *: AFF_RIGHT ID*
4. ***EXPR*** *: TERM [ARTHOP TERM]*
5. ***TERM*** *: FACT [MULTOP FACT]*
6. ***FACT*** *: VAL | "(" EXPR | ID ("," EXPR | ID) ")" | | epsilon*
7. ***AFF_LEFT*** *: AFF_LEFT INST*
8. ***SI*** *: if "(" COND ")" "{" INST "}" [else "{" INST "}" | "epsilon"]*
9. ***BOUCLE*** *: BOUCLE_FOR | BOUCLE_WHILE*
10. ***BOUCLE_WHILE*** *: while "(" COND ")" "{" INST "}"*
11. ***BOUCLE_FOR*** *: for "(" ID "in" SEQ ")" "{" INST "}"*
12. ***PRINT*** *: print "(" EXPR ")"*
13. ***INPUT*** *: readline("//msg")*
14. ***FUNCTION*** *: "function" "(" VAR_FUNCTION ")" ["{"] INST ["return" "("] VAR [")"] ["}"]*
15. ***CHIFFRE*** *: "0"|..|"9"*
16. ***LETTRE*** *: "a"|..|"z"|"A"|..|"Z"*
17. ***CHARACTER*** *: (""" | "'") [ANY]$^+$ (""" | "'")*
18. ***HEX*** *: "0"|..|"9"|"A"|..|"F"|"a"|..|"f"*
19. ***NUM*** *: $CHIFFRE^+$ | "0" ("x"|"X") HEX^+*
20. ***INTEGER*** *: $[CHIFFRE]^+$ "L"*
21. ***DOUBLE*** *: $[CHIFFRE]^+$ "." INTEGER*
22. ***COMPLEX*** *: $[DOUBLE]^+$ ("+"|"-") NUM "i"*
23. ***VECTOR*** *: EXPR ":" EXPR*
24. ***ID*** *: (".")* ("_")$^+$ (LETTRE | epsilon) $(LETTRE|CHIFFRE|"."|"\_")^*$*
25. ***COND*** *: COND1 ["||" COND1 | "" COND1]*
26. ***COND1*** *: ["("] COND2 [")"]*

27. ***COND2*** *: ["!"] COND3*
28. ***COND3*** *: COND4 [RELOP COND4]*
29. ***COND4*** *: "TRUE" | "FALSE" | EXPR*
30. ***RELOP*** *: "==" | "!=" | "<" | "<=" | ">" | ">="*
31. ***AFFOP_LEFT*** *: "<-" | "«-" | "="*
32. ***AFFOP_RIGHT*** *: "->" | "-»"*
33. ***ARTHOP*** *: "+" | "-"*
34. ***MULTOP*** *: "*" | "/" | "^" | "%%" | "%/%"*
35. ***VAL*** *: BOOLEAN | NUMERIC | INTEGER | COMPLEX | CHARACTER | RAW | VECTOR | LIST | MATRICE | ARRAY*

2.2 Tokens

1. *IF_ TOKEN (if)*
2. *ELSE_ TOKEN (else)*
3. *REPEAT_ TOKEN (repeat)*
4. *WHILE_ TOKEN (while)*
5. *FUNCTION_ TOKEN (function)*
6. *FOR_ TOKEN (for)*
7. *IN_ TOKEN (in)*
8. *NEXT_ TOKEN (next)*
9. *BREAK_ TOKEN (break)*
10. *TRUE_ TOKEN (true)*
11. *FALSE_ TOKEN (false)*
12. *NULL_ TOKEN (null)*
13. *PRINT_ TOKEN (print)*
14. *READ_LINE_ TOKEN (readline)*
15. *RETURN_ TOEKN (return)*
16. *Inf_ TOKEN*
17. *NaN_ TOKEN*
18. *NA_INTEGER_ TOKEN*
19. *NA_REAL_ TOKEN*
20. *NA_COMPLEX_ TOKEN*
21. *NA_CHARACTER_ TOKEN*
22. *ADD_ TOKEN (+)*
23. *SUB_ TOKEN (-)*

24. *MULT_ TOKEN (*)*
25. *DIV_ TOKEN (/)*
26. *EXPONENT_ TOKEN (^)*
27. *MODULUS_ TOKEN (%%)*
28. *INT_ DIV_ TOKEN (%/%)*
29. *INF_ TOKEN (<)*
30. *SUP_ TOKEN (>)*
31. *INFEG_ TOKEN (<=)*
32. *SUPEG_ TOKEN (>=)*
33. *EQUALTO_ TOKEN (==)*
34. *NOT_ EQUAL_ TOKEN (!=)*
35. *LOGICAL_ NOT_ TOKEN (!)*
36. *LOGICAL_ AND_ TOKEN (&&)*
37. *LOGICAL_ OR_ TOKEN (||)*
38. *ELEMENTWISE_ LOGICAL_ AND_ TOKEN (&)*
39. *ELEMENTWISE_ LOGICAL_ OR_ TOKEN (|)*
40. *LEFT_ ASGN_ TOKEN (<- , «- , =)*
41. *RIGHT_ ASGN_ TOKEN (-> , -»)*
42. *PO_ TOKEN (( )*
43. *PF_ TOKEN ( ))*
44. *AO_ TOKEN ({)*
45. *AF_ TOKEN (})*
46. *POINT_ TOKEN (.)*
47. *VIR_ TOKEN (,)*
48. *TWO_ POINT_ TOKEN (:)*
49. *GUIL_ TOKEN (")*
50. *APOST_ TOKEN (’)*

3. les fonctions de l'analyseur lexical

1. ***void push()*** *; Remplir la list des tokens*
2. ***void show()*** *; Afficher tokens convenable*
3. ***void Next_ Car()*** *; Lire le caractère suivant*
4. ***void Next_ Word()*** *; Lire le mot suivant*
5. ***void Next_ Number()*** *; Lire le nombre suivant*
6. ***void Next_ Character()*** *; Lire touts les caractères suivant*
7. ***void Check_ Token()*** *; Vérifier le token*
8. ***void Break_ Comment()*** *; Éviter les commentaires*
9. ***void Next_ Sym()*** *; Lire le symbole suivant*
10. ***int analy_ lex(char *filename)*** *; Analyseur lexical*

4. les fonctions pour l'analyseur syntaxique

1. ***void Symbole_ Suiv(void) ;***
2. ***void Test_ Symbole(TOKENS token,ERRORS erreur) ;***
3. ***void INSTS() ;***
4. ***void INST() ;***
5. ***void AFTER_ ID(void) ;***
6. ***void LEFT_ ASSIGN(void) ;***
7. ***void RIGHT_ ASSIGN(void) ;***
8. ***void PRINT(void) ;***
9. ***void IF(void) ;***
10. ***void BOUCLE_ FOR(void) ;***
11. ***void BOUCLE_ WHILE(void) ;***
12. ***void FUNC(void) ;***
13. ***void SEQ();***
14. ***void READ_ LINE(void) ;***
15. ***void COND(void) ;***
16. ***void COND1(void);***
17. ***void COND2(void) ;***
18. ***void COND3(void) ;***
19. ***void COND4(void) ;***
20. ***void EXPR(void) ;***
21. ***void TERM(void) ;***
22. ***void FACT(void) ;***
23. ***void CALL_ FUNC(void) ;***
24. ***int analy_ syn() ;***

5. Les règles sémantiques

TYPE_OPERATION	NUM	HEX	COMPLEX	VECTOR	MATRIX	ARRAY	CHAR
ARITHM_OPERATION	HEX COMPLEX VECTOR MATRIX	COMPLEX VECTOR MATRIX NUM	VECTOR MATRIX NUM HEX	MATRIX NUM COMPLEX HEX	NUM COMPLEX HEX VECTOR		
BOOL_OPERATION	HEX COMPLEX VECTOR MATRIX	COMPLEX VECTOR MATRIX NUM	VECTOR MATRIX NUM HEX	MATRIX NUM COMPLEX HEX	NUM COMPLEX HEX VECTOR		
LOGICAL_OPERATION	HEX COMPLEX VECTOR MATRIX ARRAY CHAR	COMPLEX VECTOR MATRIX NUM ARRAY CHAR	VECTOR MATRIX NUM HEX ARRAY CHAR	MATRIX NUM COMPLEX HEX ARRAY CHAR	NUM COMPLEX HEX VECTOR ARRAY CHAR	NUM COMPLEX HEX VECTOR CHAR MATRIX	NUM COMPLEX HEX VECTOR CHAR MATRIX
EQUAL_OPERATION	HEX COMPLEX VECTOR MATRIX	COMPLEX VECTOR MATRIX NUM	VECTOR MATRIX NUM HEX	MATRIX NUM COMPLEX HEX	NUM COMPLEX HEX VECTOR		
REL_OPERATION	HEX COMPLEX VECTOR MATRIX ARRAY CHAR	COMPLEX VECTOR MATRIX NUM ARRAY CHAR	VECTOR MATRIX NUM HEX ARRAY CHAR	MATRIX NUM COMPLEX HEX ARRAY CHAR	NUM COMPLEX HEX VECTOR ARRAY CHAR	NUM COMPLEX HEX VECTOR CHAR MATRIX	NUM COMPLEX HEX VECTOR CHAR MATRIX

Figure 5.1 – Les règles sémantique du langage R

6. Conclusions

Notre projet consiste à réaliser un compilateur du langage R en langage C. Pour atteindre cet objectif, nous avons commencé par rendre la grammaire R une grammaire LL(1).Après, nous avons commencer par réaliser un analyseur lexical.Ensuite, nous avons réaliser un analyseur syntaxique. Finalement, nous avons attaqué la partie réalisation d'un analyseur sémantique.

Ce projet de est une expérience très enrichissante. En effet, ce fût une occasion pour nous de mettre en pratique et d'élargir nos connaissances acquises à l'ENSIAS. Il nous a également donné l'opportunité d'intégrer une équipe performante et de bien connaître son métier. La réalisation du projet nous a aussi permis de raffiner nos capacités de conception et de renforcer nos compétences.