Projet Compilation

Ce projet consiste à créer un *compilateur* pour un pseudo langage C : « mini-C » et ceci afin de compiler et exécuter un *programme source avec* la *grammaire* Gf suivante :

Variables terminaux :

Vt = { entier, chaine ,ident, fmain, define, ent, lire , ecrire, si ,sinon ,tantque, affect, plus, moins, mult, divi,virg, pointvirg, parouv, parfer, inf, infegal, egal, diff ,sup, supegal, accouv, accfer, fdf }

Avec :

ident : identificateur,
fmain : fonction main,
ent : entier,
affect : affectation,
virg : virgule,
parouv : parenthèse ouvert,
parfer : parenthèse fermé,
inf : inférieur,
infegal : inférieur égale,
diff : différent,
accouv : accoulade ouverte,
accfer : accoulade fermé,
fdf : fin de fichier.

Variables non terminaux :

Vn = {FICHIER, PROGRAMME, PROGRAMME2, DECL_CONST, DECL_CONST2, DECL_VAR,DECL_VAR2, DECL_VAR3, SUITE_VAR, SUITE_VAR2, PROG, BLOC, BLOC2, AUTRES_INST, AUTRES_INST2, INSTRUCTION, CONDITIONNELLE, ITERATION, AFFECTATION, LECTURE, ECRITURE, ECRITURE2, ECRITURE3, AUTRES_ECRI, AUTRES_ECRI2, EXP_OU_CH, EXP, EXP2, TERME, OP_BIN, OP_REL}

Avec :

FICHIER: *axiome* de la grammaire.

P est l'ensemble des *règles de productions* (défini ci-dessous)

La Grammaire :

FICHIER --> PROGRAMME fdf
PROGRAMME --> DECL_CONST PROGRAMME2
| DECL_VAR PROG
| PROG
PROGRAMME2 --> DECL_VAR
| PROG
DECL_CONST --> define ident entier DECL_CONST2
DECL_CONST2 --> ε
DECL_VAR --> ent ident DECL_VAR2
DECL_VAR2 --> SUITE_VAR pointvirg DECL_VAR3
| pointvirg DECL_VAR
DECL_VAR3 --> DECL_VAR
| ε
SUITE_VAR --> virg ident SUITE_VAR2
SUITE_VAR2 --> SUITE_VAR
| ε
PROG --> fmain parouv parfer BLOC
BLOC --> accouv BLOC2
BLOC2 --> AUTRES_INST accfer

AUTRES_INST --> INSTRUCTION AUTRES_INST2
AUTRES_INST2 --> AUTRE_INST
| ε
INSTRUCTION --> CONDITIONNELLE
| ITERATION
| AFFECTATION
| LECTURE
| ECRITURE
CONDITIONELLE --> si parouv EXP parfer BLOC SUITE_COND
SUITE_COND --> ε
| sinon BLOC
ITERATION --> tantque parouv EXP parfer BLOC
AFFECTATION --> ident affect EXP pointvirg
LECTURE --> lire parouv ident parfer pointvirg
ECRITURE --> ecrire parouv ECRITURE2
ECRITURE2 --> parfer pointvirg
| EXP_OU_CH ECRITURE3
ECRITURE3 --> AUTRES_ECRI parfer pointvirg
| parfer pointvirg
AUTRES_ECRI --> virg EXP_OU_CH AUTRES_ECRI2
AUTRES_ECRI2 --> AUTRES_ECRI
| ε
EXP_OU_CH --> EXP
| chaine
EXP --> TERME EXP2
EXP2 --> OP_BIN EXP
| OP_REL EXP
| ε
TERME --> entier
| ident
| parouv EXP parfer
| moins TERME
OP_BIN --> plus
| moins
| mult
| divi
OP_REL --> inf
| infegal
| diff
| supegal
| sup

Remarques :

ε : désigne l'ensemble vide

Les Opérateurs Réels sont : **{inf , infegal , egal , diff , supegal , sup**}

Utilisez Flex et Bison pour réaliser le projet.

Ce mini-C doit permettre d'exprimer des programmes élémentaires en langage C, vous devez préparer des petits programmes test afin de les tester le jour de la soutenance. Votre programme doit tester un bout de programme en C et retourner s'il est valide ou non.

Aucun calcul à retourner, juste un test pour dire si votre programme est correct ou non.

Bonus :

Ajouter au code les commandes nécessaires pour lire un fichier qui contient le code source de votre mini C.