Mini Projet de Compilation

Le but de ce projet de compilation est d'implémenter un mini-compilateur réalisant les différentes phases de la compilation à savoir l'analyse lexicale et l'analyse syntaxico-sémantique. Les traitements parallèles concernant la gestion de la table des symboles ainsi que le traitement des différentes erreurs doivent être également réalisés lors des différentes phases d'analyse du processus de compilation.

Ainsi il vous est demandé de définir votre propre langage sur lequel le projet va porter. La définition du langage doit au minimum comporter les aspects suivants :

-Structure générale du programme.

-Commentaires.

-Déclaration de variables simples et structurées.

-Définition des différents types (appellation, plage de valeurs,…etc.).

-Instructions de base (affectation, conditions, boucles, entrée/sortie,…etc.).

-Opérateurs logiques, de comparaison, et arithmétiques.

-les différentes priorités et règles d'associativité.

L'analyse lexicale avec l'outil FLEX a pour but d'associer à chaque mot du programme source la catégorie lexicale à laquelle il appartient. Pour cela, il est demandé de définir les différentes entités lexicales à l'aide d'expressions régulières et de générer le programme FLEX correspondant. Il s'agira dans un premier temps d'implémenter les transformations vues en cours (en utilisant le C) pour générer les AFD en ayant comme input les expressions régulière du langage à analyser. Il vous est demandé de faire ce travail pour un sous-ensemble des instructions de votre langage. Dans un second temps, il vous est demandé de porter l'analyse lexicale sur l'ensemble du langage en utilisant FLEX.

L'analyse syntaxico-sémantique se base sur une grammaire syntaxique. Cette dernière est transformée en des tables d'analyse qui permette de générer un arbre syntaxique. Ces transformations sont implémentées dans l'outil BISON qui simplifie grandement cette étape. Il vous est ainsi demandé d'aborder l'analyse syntaxique en deux étapes.

Il s'agira de sélectionner une méthode (descendante ou ascendante) et d'implémenter les transformations vues en cours (en utilisant le C) pour générer les tables d'analyse syntaxiques en ayant comme input la table des symboles de l'analyse lexicale. Il vous est demandé de faire ce travail pour un sous-ensemble de la grammaire de votre langage.

Dans un second temps, il s'agira de généraliser l'analyseur syntaxique à l'ensemble de la grammaire de votre langage en utilisant l'outil BISON. Ce dernier est un analyseur ascendant qui opère sur des grammaires LALR. Il faudra spécifier dans le fichier BISON les différentes règles de la grammaire ainsi que les règles de priorités pour les opérateurs afin de résoudre les conflits. Les routines sémantiques doivent être associées aux règles dans le fichier BISON.

La génération du code intermédiaire permet de transformer le code source en ensemble d'instructions élémentaire proche du code objet sans être pour autant spécifique à une machine donnée. Il vous est demandé de baser votre transformation sur le format quadruplets.

La table de symboles doit être créée lors de la phase de l'analyse lexicale. Elle doit regrouper l'ensemble dvariables et constantes définies par le programmeur avec toutes les informations nécessaires pour le processus de compilation. Cette table sera mise à jour au fur et à mesure de l'avancement de la compilation. Il est demandé de prévoir des procédures pour permettre de rechercher et d'insérer des éléments dans la table de symboles. Les variables structurées de type tableau doivent aussi figurer dans la table.

Il est aussi demandé d'afficher les messages d'erreurs adéquats à chaque étape du processus de compilation. Ainsi, lorsqu'une erreur lexicale ou syntaxique est détectée par votre compilateur, elle doit être signalée le plus précisément possible, par sa nature et sa localisation dans le fichier source. On adoptera le format suivant pour cette signalisation : File "Test", line 4, character 56: syntax error

Il vous est demandé de vous constituer en équipe de 4 étudiants faisant partie du même groupe. Les délais prévisionnels des livrables sont définis comme suit (seront amenés à évoluer éventuellement selon l'avancement du cours).

Livrable 1 : liste des groupes	**20/11**
Livrable 2 : proposition du langage	**23/11**
Livrable 3 : livrable final	**TBA (avant examen final)**
Livrable 4 : soutenance du mini-projet	**TBA (avant examen final)**

Le livrable final contient le code, et un rapport.