TELECOM Nancy 2ème année
Octobre 2019 - Avril 2020

Projet de compilation (Modules PCL1-PCL2)

L'objectif de ce projet est d'écrire un compilateur du langage `Algol60` *(acronyme de ALgorithmic Oriented Language), créé à la fin des années 1950. Bien qu'il ne soit pas tout jeune, ce langage a été conçu comme un langage typé, procédural, à structure de blocs et permettant la récursivité.*
Votre compilateur produira en sortie du code assembleur `microPIUPK/ASM`[1]*, code assembleur que vous avez étudié dans le module PFSI de première année.*

1 Réalisation du projet et dates à retenir

Vous travaillerez par groupes de 4 élèves, et en aucun cas seul ou à deux. Si vous êtes amenés à former un trinôme, nous en tiendrons compte lors de l'évaluation de votre projet.

Vous utiliserez l'outil ANTLR, générateur d'analyseurs lexical et syntaxique *descendant*, interfacé avec le langage Java pour les étapes d'analyse lexicale et syntaxique. Vous génèrerez ensuite dans un fichier du code assembleur au format *microPIUP/ASM*.
Votre compilateur doit signaler les erreurs lexicales, syntaxiques et sémantiques rencontrées. Lorsqu'une de ces erreurs lexicales ou sémantique est rencontrée, elle doit être signalée par un message relativement explicite comprenant, dans la mesure du possible, un numéro de ligne. Votre compilateur peut s'arrêter après chaque erreur syntaxique détectée (pas d'obligation de reprise). En revanche, votre compilateur doit impérativement poursuivre l'analyse après avoir signalé une erreur sémantique.

Vous utiliserez un dépôt Git sur le Gitlab de TELECOM Nancy[2]. Créez votre projet dans votre espace personnel. Il devra être privé et l'identifiant sera de la forme login1 (où login1 est le login du membre chef de projet de votre groupe). Vous ajouterez Suzanne Collin, Sébastien Da Silva et Gérald Oster en tant que "Master" de votre projet. Votre répertoire devra contenir tous les fichiers sources de votre projet, les dossiers intermédiaire et final (au format PDF) ainsi qu'un *mode d'emploi* pour utiliser votre compilateur.

Les dépots seront régulièrement consultés par les enseignants chargés de vous évaluer lors des séances de TP et lors de la soutenance finale de votre projet.
En cas de litige sur la participation active de chacun des membres du groupe au projet, le contenu de votre projet sur le dépôt sera examiné. Les notes peuvent être individualisées.

Les modules PCL1 et PCL2 (qui ne font pas partie du module TRAD1) sont composés de 7 séances de TP. Vous serez évalués à plusieurs reprises :

- semaine 46 (12 au 15 novembre), lors de la séance TP3 au cours de laquelle vous nous présenterez la grammaire et l'AST. Vous rendrez pour le 15 janvier un premier dossier (cf. rubrique "Les dossiers" pour en connaître le contenu). Cette évaluation et le dossier comptent pour la validation du module PCL1.
- semaine 7 (10 au 14 février), vous nous montrerez lors de la séance de TP les contrôles sémantiques réalisés ainsi que la TDS complète.
- en fin du module PCL2 (le 27 avril 2020), vous présenterez lors d'une soutenance le fonctionnement de votre compilateur dans sa version finale, comprenant donc la génération de code assembleur. Vous rendrez aussi à la fin du projet, un dossier qui entrera dans l'évaluation de votre projet.

[1] Le jeu d'instructions et son codage ont été définis par Alexandre Parodi et la syntaxe du langage d'assemblage par Karol Proch.
[2] https://gitlab.telecomnancy.univ-lorraine.fr/

Déroulement des modules PCL1 et PCL2

PCL1 : séances TP 1 à 4 - Prise en main du logiciel ANTLR, définition complète de la grammaire du langage et de l'AST, début de TDS.

On vous propose une initiation au logiciel ANTLR lors de la première séance. Vous définirez ensuite la **grammaire LL(1)** du langage et la soumettrez à ANTLR afin qu'il génère l'analyseur syntaxique descendant. Bien sûr, l'étape d'analyse lexicale est réalisée parallèlement à l'analyse syntaxique.
Vous aurez testé votre grammaire sur des exemples variés de programmes écrits en `Algol60` (avec et sans erreurs lexicales et syntaxiques).
Vous réfléchirez ensuite à la construction de l'arbre abstrait que vous mettrez en oeuvre.
Comme mentionné précédemment, une démonstration de l'AST (construit à partir d'une grammaire LL(1)) sera faite lors de la séance TP3.
Vous commencerez alors à réfléchir à la mise en oeuvre de la TDS.

PCL2 : séances TP 1 et 2 - Finalisation de la construction de la TDS, contrôles sémantiques.

A partir de la rentrée (semaine 3), vous terminez la construction de la table des symboles, et vous implémentez les contrôles sémantiques. A la fin de cette itération (c'est à dire en séance TP2 - semaine 7 - du 10 au 14 février), vous montrerez la table des symboles finale (une visualisation, même sommaire, est indispensable) ainsi que les contrôles sémantiques implémentés. Vous serez évalués et vous obtiendrez une note pour cette démonstration.

Fin du projet : génération de code assembleur.

Vous continuez votre projet par l'étape de génération de code. Pour cette dernière phase, vous veillerez à générer le code assembleur de manière incrémentale, en commençant par les structures "simples" du langage.

Votre projet sera testé par les enseignants de TP et se fera en votre présence. Il est impératif que vous ayiez prévu des exemples de programmes permettant de tester votre projet et ses limites (ces exemples ne seront pas à écrire le jour de la démonstration) : vous pourrez nous montrer votre "plus beau" programme...

Les dossiers

1. A la fin du module PCL1, et pour le 15 janvier 2020, vous rendrez un premier rapport présentant :

   - la grammaire du langage,
   - la structure de l'arbre abstrait,
   - des *jeux d'essais* illustrant le bon fonctionnement de votre grammaire et l'AST construit.
   - les éléments de gestion de projet relatifs à cette première partie du projet (fiche projet, CR rédigés, Gantt, fiche d'évaluation de la répartition du travail, répartition des tâches au sein du groupe).

2. A la fin du projet (donc à la fin du module PCL2) et pour le 27 avril 8h, vous rendrez un dossier qui complètera le précédent et comprendra *au moins* :

   - la structure de la table des symboles que vous avez définie,
   - les erreurs sémantiques traitées par votre compilateur,
   - les schémas de traduction (du langage proposé vers le langage assembleur) les plus pertinents,
   - des *jeux d'essais* mettant en évidence le bon fonctionnement de votre programme (erreurs correctement traitées, exécutions dans le cas d'un programme correct), et ses limites éventuelles.
   - une partie *gestion de projet*, à savoir une fiche d'évaluation de la répartition du travail sur cette seconde période avec la répartition des tâches au sein de votre binôme, l'estimation du temps passé sur chaque partie du projet, et le Gantt final.
   - les divers CR de réunion que vous avez rédigés.

   Vous remettrez ces dossiers dans le casier de votre enseignant de TP.

Pour finir...

Bien entendu, il est interdit de "s'inspirer trop fortement" du code d'un autre groupe ; vous pouvez discuter entre-vous sur les structures de données à mettre en place, sur certains points techniques à mettre en oeuvre, etc... mais il est interdit de copier du code source sur vos camarades. Si cela devait se produire, nous saurons en tenir compte dans nos évaluations.

Rappel : les notes peuvent être individualisées.

La fin du projet est fixée au **lundi 27 avril 2020**, date prévue pour les soutenances finales ; lors de cette soutenance, on vous demandera de faire une démonstration de votre compilateur. Un planning vous sera proposé pour fixer l'ordre de passage des groupes.

Aucun délai supplémentaire ne sera accordé pour la fin du projet. Le temps restant sur le mois de mai est réservé à la finalisation de vos autres projets et du PIDR.

2 Présentation du langage

Les liens suivants contiennent l'ensemble des spécifications du langage Algol60 et de nombreux exemples de programmes écrits dans ce langage. On vous demande d'écrire la grammaire, de construire l'AST et d'effectuer les contrôles sémantiques pour le langage complet.

`http://www.algol60.org`
`http://www.algol60.org/6legoPieces.htm`
`https://www.masswerk.at/algol60/`

Concernant la génération de code, on vous fournira 4 "niveaux" de programmes écrits en Algol60, ce qui vous permettra de vous situer dans cette phase de génération de code. Le niveau 2 est généralement celui requis pour obtenir une note de 10/20. Bien entendu, ce n'est qu'une indication, car la note finale dépend aussi des évaluations intermédiaires et des rapports rendus.

3 Génération de code

Le code généré devra être en langage d'assemblage *microPIUPK/ASM* écrit dans un fichier texte d'extension `.src` , en utilisant un sous-ensemble des instructions de la machine.
Le fichier généré devra être assemblé à l'aide de l'assembleur qui générera un fichier de code machine d'extension `*.iup`.
Ce dernier sera exécuté à l'aide du simulateur du processeur `APR`[3].
Ces deux outils (assembleur et simulateur) fonctionnent sur toute machine Windows ou Linux disposant d'un runtime java. Ils sont inclus dans le fichier (archive java) `microPIUPK.jar` disponible sur le serveur neptune dans le dossier `/home/depot/PFSI`.

Ce fichier supporte les commandes suivantes, en supposant que le fichier `microPIUPK.jar` soit dans le dossier courant.

1. Assembler un fichier `projet.src` dans le fichier de code machine `projet.iup` :
   `java -jar microPIUPK.jar -ass projet.src`
2. Exécuter en batch le fichier de code machine `projet.iup` :
   `java -jar microPIUPK.jar -batch projet.iup`
3. Lancer le simulateur sur interface graphique :
   `java -jar microPIUPK.jar -sim`

[3] Advanced Pedagogic RISC développé par Alexandre Parodi qui comporte toutes les instructions et modes d'adressage pour permettre l'enseignement général de l'assembleur, mais dont un sous-ensemble forme une machine RISC facilitant l'implémentation matérielle sur une puce et (on l'espère) l'écriture d'un compilateur.

Cette section concerne les élèves de 3A aménagée

Vous êtes 4 élèves à repasser ce module, vous formerez donc 1 seul groupe. La fin du projet est fixée fin février, avant votre départ en stage. Vous serez évalués par une démonstration finale de votre compilateur, la remise d'un rapport final (pour le contenu, cf. les sections précédentes), ainsi qu'une évaluation début janvier 2020 impérativement, de la grammaire, AST, TDS et contrôles sémantiques réalisés. Un premier rapport sera remis à cette date, décrivant les structures mises en oeuvre, jeux d'essais, etc, sans oublier la partie *gestion de projet*. **Cette évaluation ne peut être reportée car elle entre dans la validation du module PCL1.**

Vous n'êtes pas tenus d'assister aux séances de TP.

Cette section concerne les élèves de 4A

Vous êtes 8 élèves à repasser ce module, vous formerez par conséquent 2 groupes. La fin du projet est fixée fin février. Vous serez évalués par une démonstration finale de votre compilateur, la remise d'un rapport final (pour le contenu, cf. les sections précédentes, n'oubliez pas la partie *gestion de projet*), ainsi qu'une évaluation de la grammaire, AST, TDS et les contrôles sémantiques réalisés. Vous nous montrerez ces premiers résultats **dès qu'ils seront implémentés**.

Vous n'êtes pas tenus d'assister aux séances de TP.