PROJET DE COMPILATION :
Rapport

Auteurs:
Adrien Fievet
Claire D'Haene

2023-2024

Titulaire
Véronique BRUYERE
Alexandre DECAN

Etudiants
Adrien (220625)
Claire (220323)

Table des matières

1 Introduction

Dans le cadre de notre cours de compilation, nous devons réaliser un projet. Ce projet a pour but de mettre en oeuvre les connaissances acquises durant le cours ainsi que notre capacité à nous documenter afin de concevoir un interpréteur pour le langage SPF (Simple Programme en Français) que nous allons décrire lors des prochaines sections. Lark et python seront utilisés pour implémenter ce projet.

2 Grammaire implémentée

D'abord, nous avons conçus notre programme comme un nombre indétermimé d'instructions. Cette dernière se définisant par soit une assignation, un affichage, un ajout, une condition, une boucle ou encore une expression. Le " ;" n'a pas été défini globalement mais au cas par cas car les conditions et les boucles sont délimitées par des accolades. Nous allons passer en revu chacun de ces instructions :

2.1 Les déclarations

Une déclaration est simplement défini par un type, une variable ainsi que potentiellement initialisée avec une expression. Le langage SPF contient quatres types :

— booléen
— entier
— texte
— liste

Notez qu'une liste est défini comme une liste de valeur peu importe le type ou alors par un séquence de nombre entier. Une variable quant à elle doit respecter la grammaire suivante : Le nom d'une variable est composé au minimum d'un caractère, ne peut contenir que des lettres (majuscules et minuscules) accentuées ou non, des chiffres ou un tiret bas. Le nom d'une variable ne peut pas débuter par un chiffre. Nous avons donc obligé un premier terme avec une lettre ou un underscore et ensuite un nombre indétermimé de lettre, de chiffres et de underscore. Nous reparlerons de la grammaire d'une expression

2.2 Les assignations

Elles sont très similaires aux déclarations mise à part le faite que le type ne doit plus être précisé mais maintenant, il faut obligatoirement une expression.

2.3 L'affichage

afficher prend simplement une suite d'au moins une expression séparées par des virgules. Nous nous sommes également permis de rajouter la possibilité d'afficher directement le résultat de l'instruction **ajout**.

2.4 L'ajout

Pour **ajout**, nous définissons une grammaire avec une expression et une variable dans laquelle sera rajoutée le résultat de l'expression.

2.5 Les boucles

boucle représente simplement les deux possibilités de boucle offertes par le langage SPF. À savoir **tantque** qui prend une expression et une suite d'instruction. Et **pourchaque** qui a également besoin d'une variable et de son type.

2.6 Les conditions

Tout comme le point précédent, **condition** contient une règle **si** qui exécute les instructions en fonction de la condition. Et **sisinon**, une extension de **si** qui prend en plus d'autres instructions dans un autre corps d'accolades.

2.7 Les expressions

Les expressions se définissent de base comme des litéraux ou des opérations mais peuvent être englobées par des parenthèses pour émettre des priorités ou encore être représentées par une variable.

2.7.1 Les litéraux

Les litéraux représentent toutes les valeurs possibles par rapport aux quatres types de ce langage.

Les booléens : Valant soit *vrai* soit *faux*.

Les entiers : Une séquence de chiffre ne commencant pas par 0 sauf si c'est pour représenter 0. Il peut optionnellement avoir un *plus*.

Les textes : Une suite de n'importe quels symboles entre deux *guillemets*.

Les listes : Soit une liste d'expression entre *crochets* séparées par des *virgules*. Soit une sequence définit par deux expressions entre *crochets* séparées par un *deux-point*.

2.7.2 Les opérations

2.8 Autres

Nous avons également définit une règle pour ignorer tous les espaces blancs et une règle pour ignorer tous les caractères après un *dièse* sur une ligne.

3 Notre approche

3.1 Les variables

3.2 Le test conditionnel

3.3 Les boucles

4 Les erreurs connues et les solutions envisagées

5 Difficultés rencontrées

6 Répartition du travail

En ce qui concerne la répartition des tâches, dans un premier temps, nous fonctionnons en se documentant chacun de notre côté et ensuite, nous rassemblons nos idées. Au niveau de l'implémentation, cela fonctionne de la même manière. De plus, nous prévoyons beaucoup d'appels et de séances en présentiel afin d'avancer à deux sur le projet. Ceci nous permettant donc d'avancer plus rapidement et efficacement.

7 Conclusion