Analyse syntaxique de la machine de Turing +

Kenza Liza

Alice

Objectif

input : Prend en entrée un fichier .TSplus

- Vérifier la syntaxe de la machine de Turing de del Vigna + et afficher des messages d'erreur s'il y a des erreurs dans le script spécifié. Si pas d'erreur, affiche "Fichier Valide".

- Crée en sortie, si la syntaxe est correcte, un fichier TSV spécifiant la liste des tokens, leur type, l'instruction à laquelle ils appartiennent, le type de l'instruction, et leur position dans l'instruction s'il s'agit d'une instruction de type "opération" ou "test"..

Tokens autorisés

Dans MTdV:

- G, D, 0, 1, fin, si (0), si (1), boucle, }, #, I, P, % (commentaire)

Dans MTdV+:

- entiers [0-29]
- nom de variable ([a-z]|[A-Z]|_)([0-9]|[a-z]|[A-Z]|_)* : un nom de variable peut contenir des chiffres, des lettres minuscules ou majuscules et le symbole _, mais ne peut pas commencer par un chiffre. De plus, il ne peut pas être un autre mot clé (ex boucle = 3).
- mot clé ‘si’ → pour faire des test sur les valeurs et les variables.
- opérateurs : =, +, *, == , nous envisageons l’ajout de: <, >, - (avec erreur si négatif mais c’est le problème du programmeur, pas le nôtre)

Type des tokens

- fin, si (0), si (1), boucle, }, #, I, P, % : “**trivial**”

  => il s’agit de tokens qui ne changent pas de sens par rapport à la MTdV

- G, D, 0, 1 : “**mémoire**”

  => tokens qui touchent à la mémoire.

- si : “**complexe**”
- x, y, ma_variable : “**var**”
- +, *, =, == : “**op**”
- [0-29] : “**val**”

  => afin de savoir s’il s’agit de “val” ou “mémoire” pour 0 et 1, il faut regarder le contexte du token (désambiguisation)

Nature des erreurs

Dans MTdV, nous avions les erreurs suivantes:

- Mauvais chemin vers le fichier (404 Not Found :( )
- Token erroné : *token* (ligne n)
- Terminé sans avoir rencontré de #
- Tokens trouvés après le caractère final # : *token_suivant* (ligne n)
- } rencontré hors d'une *boucle* ou d'un *si* (ligne n)
- Pas d'accolade fermante après *si* (ligne n)
- Pas d'accolade fermante après *boucle* (ligne n)
- Pas de *fin* dans la *boucle* (ligne n)

Ainsi que le warning:

- *fin* rencontré à l'extérieur d'une boucle, le programme pourrait s'arrêter avant la fin.

Nature des erreurs

Nous allons ajouter les erreurs suivantes:

- Opération binaire incomplète (il manque un des trois : *gauche*, *opérateur*, *ou droite)* (ligne n*)*
- Affectation incomplète (*variable*, *=*, *partie de droite)* (ligne n*)*
- Entier dépassant la limite autorisée (<0 ou > 29) : *entier* (ligne n*)*
- Token non autorisé à gauche de l'affectation : *token* (ligne n)
- Token non autorisé à droite de l'affectation : *token* (ligne n)

Ainsi que le nouveau warning :

- Mot-clé [G D 1 0] rencontré (ligne n). Cela pourrait corrompre la mémoire du programme.

Nature des erreurs

L’erreur “Opération binaire incomplète (il manque un des trois : *gauche, opérateur, ou droite)* (ligne n)”, sera renvoyée dans les cas suivants :

- 3 3 (manque opérateur)
- 3 + (manque droite)
- + 3 (manque gauche)
- + (manque droite et gauche)

Exemple d'output

n°instruction	token	type_token	type_instruction	position_affectation
0	x	var	affectation	G
0	=	op	affectation	M
0	1	val	affectation	D
1	si(0)	trivial	test	
2	si	complexe	test	
2	x	var	test	G
2	==	op	test	M
2	3	val	test	D

Exemple de programme valide

```
% Script basique MTdV+
x = 0
x = x + 1
y = x * 2
boucle
    si (x == y)
    fin }
    x = y
}
#
```

Output qui va être passé aux autres groupes

id_instruction	token	type_token	type_instruction	position_operateur
0	%	trivial	MTdV	-
1	x	variable	affectation	G
1	=	operateur	affectation	M
1	=	valeur	affectation	D
2	x	variable	affectation	G
2	=	operateur	affectation	M
2	x	variable	affectation	D
2	+	operateur	affectation	D
2	1	valeur	affectation	D
3	y	variable	affectation	G
3	=	operateur	affectation	M
3	x	variable	affectation	D
3	*	operateur	affectation	D
3	2	valeur	affectation	D
4	boucle	trivial	MTdV	-
4	si	complexe	si	-
4	x	variable	test	G
4	==	operateur	test	M
4	y	variable	test	D
4	fin	trivial	MTdV	-
5	}	trivial	MTdV	-
6	x	variable	affectation	G
6	=	operateur	affectation	M
6	y	variable	affectation	D
7	}	trivial	MTdV	-
8	#	trivial	MTdV	-


G1
G2
G3
G4
15 minutes de pause !