Document de validation

Equipe 10

January 15, 2021

1 Descriptifs de tests

1.1 Tests par section

1.2 Organisation des tests

1.3 Objectifs des tests

2 Scripts des tests

3 Gestion des risques et des rendus

3.1 Risques

Pour la partie A, en particulier l'analyse lexicale, le principale risque de bugs est que des mauvais TOKENS soient reconnus, ceci dû à une mauvaise expression régulière qui pourrait mener à une structure de l'arbre erronée.
Du côté de l'analyse syntaxique, il peut s'agir d'une instruction (setLocation par exemple) non existante ou mal placée qui pourrait mener également à une mauvaise structure de l'arbre.

Pour la partie B, il s'agit surtout de contresens avec le sujet, par exemple une mauvaise compréhension du type String.

Enfin, pour la partie C, les risques proviennent principalement du code assembleur. Il se peut que le code généré est erroné, menant à une erreur. Toutes les erreurs possibles ne sont pas forcément levées également, par exemple on peut avoir un nombre non stockable sur 32 bits en sortie de multiplication sans que cela soit relevé, menant à des bugs. Il est aussi possible que les registres et les piles ne soient pas optimalement gérés, ce qui peut donner des calculs erronés.

Avec le Git partagé, il y a également possibilité de bugs à force d'ajouter et de modifier les fichiers dans notre dépôt.

3.2 Limitation des risques

Afin de limiter les risques techniques, la partie test de l'implémentation est plutôt conséquente dans notre agenda, environ un jour ou plus afin de tester le lexer, le parser... Afin d'éviter tout bug lié à la grammaire, nous essayons les tests valides et invalides afin de bien vérifier que ce qui devrait fonctionner fonctionne et vice versa.

Le code est testé avec les scripts *test_synt*, *test_lex* ... sur les tests fournis et ceux que nous avons créé en complément.

Pour le problème du dépôt, il est possible de push uniquement les fichiers modifiés et testés afin d'éviter la création de bugs.

3.3 Gestion des rendus

En terme de rendu, nous nous organisons avec un diagramme de Gantt, avec chacun ses tâches de manière à être à l'heure pour les rendus. Pour le moment, nous n'avons pas de problème de retard.

3.4 Gestion des mises en production

En aval du rendu, afin de compléter les tests effectués, nous pouvons relire le code des autres afin de s'assurer qu'aucune erreur non visible aux tests ne s'y glisse.

4 Résultats de Jacoco

5 Méthodes de validation autre que les tests