LOG3430 – Méthodes de test et de validation du logiciel

Laboratoire 4

Tests basés sur les états

Département de génie informatique et de génie logiciel

École Polytechnique de Montréal



Soumis par

Roman Zhornytskiy (1899786)

Hakim Payman (1938609)

Gabriel Tagliabracci (1935775)

Groupe : 02

Soumis à Noureddine Kerzazi

Hiver 2020

4.1. Construire le diagramme d’états de l’interprétateur « Parcer\_FSM.py » et donner à chaque transition un nom, à chaque état également.

Dans le diagramme suivant, nous considérons que :

« Character » représente l’expression régulière suivante : "[A-Za-z|+|-|\d]"

A screenshot of a cell phone

Description automatically generated

Figure : Diagramme d'états de "Paser\_FSM.py"

4.2. Construire l’arbre des transitions de l’interprétateur « Parcer\_FSM.py ». Créer une table de transition des changements d'état en explorant le code et proposez de nouvelles transitions possibles.

Dans l’arbre de transitions suivant, nous considérons que :

« Character » représente l’expression régulière suivante : "[A-Za-z|+|-|\d]"

A close up of text on a white background

Description automatically generated

Figure : Arbre de transitions de Parser\_FSM.py

4.3. Identifiez tous les cas de tests avec les conditions à partir de l’arbre trouvé.

4.4. À l’aide de Unittest, écrire une classe de test unitaire pour tester les cas de test identifiés dans la question précédente.

4.5. À l’aide de l’outil Coverage.py, évaluez la couverture de l’interprétateur « Parcer\_FSM.py » selon la couverture des branches et identifiez les branches non couvertes, s’il y en a.