Projet d’informatique rapport de livrable 1 :

Rimbaud Sébastien
Jaoua Donia

Nous allons dans ce rapport présenter les aspects de notre projet, nos difficultés, les aspects qui fonctionnent et ceux qui ne fonctionnent pas du projet ainsi que les raisons de leurs dysfonctionnement.

I/ Objectifs :

a) But :

L’objectif du projet était d’implémenter les fonctions sfs_read et sfs_print qui permettent respectivement de lire une entrée utilisateur et de créer l’objet correspondant à l’entrée et d’afficher un objet.

b) Démarche :

Pour résoudre ces problèmes nous avons établi une démarche incrémentale et nous avons testé nos fonctions à chaque implémentation (dans la mesure du possible). Nous avons mis en place des fonctions permettant de manipuler nos structures de données (object_t et num) et nous manipulons ces structures uniquement par ces fonctions. Une opération sur notre structure de donnée est traduite par une fonction ayant la convention de nommage suivante : STRUCTURE_action_realisee(<arguments>).
Nous avons également essayé de tenir à jour une documentation de notre code source afin de facilement le communiquer et de pouvoir aisément le retravailler par la suite en accédant rapidement aux informations clés.

II/ Opérationnel :

Au terme du premier livrable notre interpréteur est capable de générer des objets à partir de différentes entrées de l’utilisateur. Il comprend les entiers signés ou non, les booléens, les caractères , les chaînes de caractères ainsi que les listes et les symboles. Sur les 13 tests proposés par défaut et dis « simple » notre projet réussi à passer les 13 avec succès. De plus nous passons 24 des 26 tests dit « évolués ».
Pour le test 30 strange_string, il y a un cas pour lequel une question est soulevée. Lorsque l’utilisateur entre une chaîne contenant un backslash suivi d’une guillemet devons-nous afficher seulement la guillemet ou le backslash et la guillemet ?
D’après la R5RS le backslash est un caractère qui protège une guillemet ou un backslash et en annule l’interprétation. De ce fait nous avons compris qu’il ne fallait pas afficher le backslash lorsqu’il protège un caractère dans un print. Pour tous les autres caractères un backslash est considéré comme caractère à part entière même s’il est seul (la R5RS ne précise pas de comportement dans ce cas).

Il y a une détection des erreurs dans l’entrée de l’utilisateur pour tous les types à l’exception des chaînes de caractères ou la seule erreur possible est de mettre les guillemets au mauvais endroit ce qui est géré par sfs_get_Sexpression et read_pair pour les mêmes raisons.
Pour ce qui est des autres fonctions celles-ci vérifient que l’entrée est valide. Pour un booléen on vérifie que celui-ci à le format #t ou #f, #r est rejeté ainsi que #tr, il en va de même pour les caractères et les entiers.
Les entiers supportent également la notation infinie.

Pour les symboles il y a un contrôle de tous les caractères admissibles dans un symbole selon la R5RS. Une fonction (isealnum) a été implémentée dans cet unique but, elle appelle isalnum et vérifie également des caractères supplémentaires spécifiés par la R5RS.
Les fonctions permettant de manipuler la structure num possèdent déjà des fonctionnalités inhérentes au flottant ce qui permettra une implémentation rapide de ceux-ci dans les futurs livrables.

II/ Non fonctionnel :

Entrer un symbole ou une chaîne de caractère trop grand ne fonctionne pas. Cela provient du fait que nous travaillons avec des chaînes de taille fixe et que nous n'allouons pas dynamiquement les chaînes de caractères selon nos besoins. Implémenter cette fonctionnalité demandera un peu plus de travail étant donné que nous devons créer de nouvelles fonctions de manipulation des strings. De plus les fonctions de lecture des symboles et des chaînes doivent être repensées pour l'ajout de cette fonctionnalité.

III/ Les points difficiles :

Les points les plus difficiles provenaient de la gestion des erreurs. En effet tenir compte des erreurs commises par l'utilisateur complique le code. Nous avons également eu du mal à comprendre les raisons du non fonctionnement d'un script de test. Pour le booléens nous n'avions pas immédiatement compris qu'il fallait faire crasher l'interpréteur malgré la bonne gestion des erreurs.