Sushi++ - Language grammar and description

COMPILERS

Floriane Magera (S111295)
Romain Mormont (S110940)
Fabrice Servais (S111093)

March 6, 2015

1 Introduction?

2 Description

Sushi++ is a mostly functional language, enhanced with a few features belonging to the imperative paradigm.

2.1 Features

Here is our list of features presented in a decreasing order of priority:

1. Anonymous function,
2. Closure: a function will be able to capture a variable or a function from a higher scope,
3. Delimiter of end of line: \n (...),
4. Full type inference,
5. Type hinting: allow the user to specify the type of a function parameter,
6. Built-in datastructures: array, list and tuple,
7. Usage of language C to use a few function,
8. Simple garbage collector: to clean the memory after the usage of built-in datastructures.
9. Packages: functions could be declared in namespaces/packages.

2.2 Keywords

The keywords of this language are inspired from the lexical field of sushi food. We have:

- `maki`: declaration of a variable or a named function.
- `soy`: declaration of an anonymous function.
- `roll`: while loop.
- `for`: for loop.
- `continue`, `break`: loop iteration control.
- `if-elseif-else`: conditional structure.
- `menu`: switch structure.
- `nori`: return.
- `to`: list "constructor".
- Types: `int`, `float`, `string`, `array`, `list`, `tuple`
- `mat`: define a package. This is not taken into account yet, but will be considered if time allows it.

2.3 Built-in datastructures

We consider to implement 3 types of datastructure of which the behavior will be defined based on 3 properties:

- **Structure mutability**: it is mutable if items can be added and removed after the creation of the datastructure,
- **Item mutability**: it is mutable if the items can be changed after the creation of the datastructure,
- **"Multityping"**: if the datastructure can contain elements of different types.

Name	Structure mutability	Item mutability	Multityping
`array`	Mutable	Mutable	Single type
`list`	Immutable	Immutable	Single type
`tuple`	Immutable	Mutable	Multiple types

Table 1: Properties of the datastructures

2.4 Operators

2.5 Functions

A function can be either a named function or an anonymous function. The structure is defined in the grammar, with the "decl-func" rule.