LINGI2132 - Languages & Translators
Part 1: Grammar & AST

Group AC
d'Oultremont Augustin - 2239 1700
Lemaire Valentin - 1634 1700

March 12, 2021

1 Introduction

We've decided to call our language ”Natural Sequencer”, with its file extension `.ns`. The syntax of this programming language is heavily inspired from python.

2 Features

2.1 General Features

We made the choice to have a dynamically typed language, therefore variables do not have explicit type declarations. We also chose not to have a separate declaration and initialization of variables meaning that variables must be assigned their starting values by the user, therefore there are no default values for variables.

2.2 Types

The language is dynamically typed. The keyword used to describe an empty value is `None` and booleans are, like in python `True` and `False`. Integers can only be expressed in decimal and floats are not implemented. Strings use double quotes to be expressed (`"Hello World"`).

Arrays are lists of values that cannot be resized. They can either be defined explicitly (`[1, 2, 3]`) or can be created empty with a certain size (`[:n]`). Map objects are dictionaries associating one value to another ({`"Hello" : 1, "World" : 2`}).

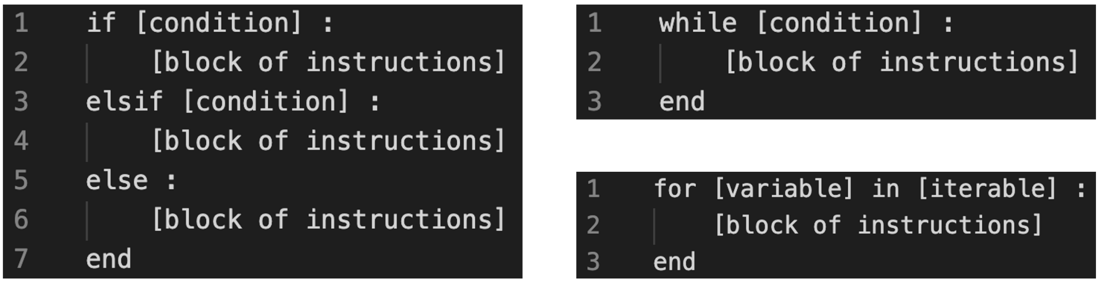
2.3 Operations & Comparisons

Booleans can be used with `or`, `and` and `not`. Values can be compared (`==`, `!=`, `<=`, `>=`, `<` and `>`). Operations on integers are : negation (using `-`), addition (`+`), subtraction (`-`), multiplication (`*`), division (`/`) and modulo (`%`) with usual priorities of operations. Braces can be used to group operations and bypass the usual priority of operations (`2*(1+2) = 6`) A particularity of our language is that the negation has priority over regular addition and subtraction (i.e. `1 + -2` is a valid expression).

2.4 Conditions & Loops

No separation between two statements is needed, except for at least one character that counts as white space (`" "`, `"\t"`, `"\n"`) i.e. we can write `x = 1 a = 2` on the same line. A block of statements is ended with the `end` keyword. Conditional blocks are done with the `if`, `elsif` and `else` keywords, following the syntax shown in figure 1 where the `elsif` and `else` blocks are optional.

The while and for loops are defined by the `while` and `for` keywords and follow the syntax shown in figure 1. The for loop iterates over a list of elements, similar to python's for loop.

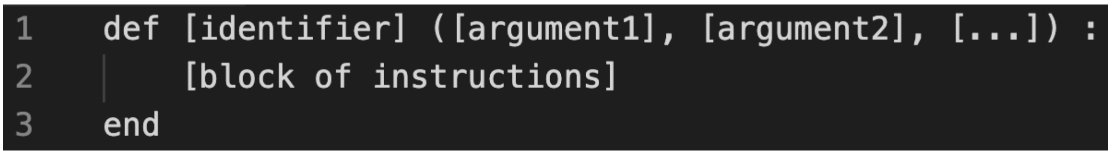


```
if [condition] :
    [block of instructions]
elsif [condition] :
    [block of instructions]
else :
    [block of instructions]
end
```

```
while [condition] :
    [block of instructions]
end
```

```
for [variable] in [iterable] :
    [block of instructions]
end
```

Figure 1: `if`, `while` and `for` examples



```
def [identifier] ([argument1], [argument2], [...]) :
    [block of instructions]
end
```

Figure 2: function example

2.5 Identifiers & Functions

An identifier (for a variable or a function) is a string that starts with a letter (lower or uppercase). A variable assignment is done with a simple equal character (`[identifier] = [value]`) and a function is defined by the keyword `def`, followed by an identifier, then the arguments between braces and separated with a comma (as showed in the example of figure 2). The function can then be called using its identifier, followed by braces containing the arguments. The function can return a value with the `return` keyword.

Some base functions can be used :

- `range(n)` returns a list of the n first integers (`[0, 1, ..., n-1]`).
- `len(l)` will return the length of the list `l`.
- `int(s)` will parse the value in `s` and return the corresponding integer.
- `print(v)` and `println(v)` will print the value `v` to the standard output (`println` adding a new line at the end).
- `sort(l)` will return a sorted copy of the list `l`.
- `indexer(a)` is a function that returns a list of indexes for an array or a map. If `a` is an array it is equivalent to `range(len(a))`, but if `a` is a dictionary, it will return a list of all keys of the dictionary, allowing the user to iterate over the dictionary.

2.6 Comments

All the text between a `#` character and a new line (or the end of the file if there is no new line) will not be considered by the compiler.