PyLit

A (very) lite version of Python.

Ryan LoPrete (rjl2172)

September 19, 2018

I. Introduction and Description

Scripting languages are becoming increasingly popular even for non-scripting applications, such as building APIs. Often times they emphasize code readability while offering multiple programming paradigms, and have extensive package support for a variety of use cases. As the fan base of these languages continues to grow, we're likely to see additional languages emerge that provide even more robust capabilities. This will allow programmers to write cleaner, more efficient, and more maintainable code. It provides motivation for the creation of a high-level programing language which is immediately purposeful, inherently.

PyLit aims to provide a simple, lightweight implementation of the common scripting language, Python. The language will have many of the basic data structures that are native to Python, such as dictionaries, strings, lists, and tuples – which will allow the user to create abstractions. The data types will support common operations such as retrieving the first element in a list, or pushing a value to a dictionary. Indentation to signify code blocks will be replaced by brackets, as used in most other languages.

The language will support function definitions using the 'def' keyword as is done in Python.

II. Syntax

Table 1. Reserved Keywords

`and`	`for`	`not`
`def`	`if`	`print`
`else / elif`	`in`	`return`

Table 2. Common Operators

Operator	Example(s)
Assignment	array = [], string = "Hello!"
Comparison	x==y, x!=y, x>y, x<y, x>=y, x<=y
Concatenation	"Hello" + "World!"
Logical (boolean)	if not x, if x and y
Mathematical operations	x+y, x-y, x*y, x/y

Table 3. Creating Data Structures

Data Structure	Example(s)
Dictionary	d = {"color": "red"}
List	array = ["a", "b", "c"]
String	string = "PyLit!"
Tuple	tuple = ("Hello", 5, "World")

Comments will be denoted by # (i.e. `#This is a comment)`.

III. Sample Program

```
# Find max value in an array

def find_max (MyArray) {
max = 0
for x in MyArray {
    if x > max {
        max = x
        }
    }
return max
}


# GCD algorithm using recursion

def gcd(x, y) {
    if y>x {
        return gcd(y,x)
    }
    if x%y == 0 {
        return y
    }
    return gcd (y, x%y)
}
```