

Project #5
CpSc 8270: Language Translation
Computer Science Division, Clemson University
Python Functions, Scope & Decision
Brian Malloy, PhD
November 15, 2018

Due Date:

In order to receive credit for this assignment, your project must be submitted, using the `web handin` command, by 8 AM, Monday, December 3rd of 2018. If you are unable to complete the project by the first due date, you may submit the project within three days after the due date with a ten point deduction.

Project Specification:

1. Your solution should translate the constructs from the previous project, including integer, float, and string values; variables; print; assignment; and the expressions specified in the previous project.
2. In addition, your solution should translate Python functions, including scope resolution 1a, return value propagation 1b and 1c.
3. In addition, your solution should translate `if/elif/else`. You must also translate the six (6) relational operators: `<`, `<=`, `==`, `>`, `>=`, `!=`. You are not required to implement `and`, `or`, `not`.
4. In all cases, the oracle for correctness is a Python 2.7 interpreter; your expressions should evaluate to the same value as a Python 2.7 interpreter, but not the same format. So, `5` is the same as `5.0`; `True` is the same as `1`, `False` is the same as `0`.
5. In the directory that contains your working interpreter, place a new directory titled `cases` that contains test cases that adequately test your interpreter.
6. Write a test harness, `test.py`, and place it in your project folder so that it runs the test cases in `cases`.
7. Your code should be well organized, formatted, readable, leak and warning free, and exploit object technology.

Light at the end of the tunnel:

In the final project, Project #6, we will translate actual and formal parameters and recursion.

<pre>def f(): x = 0 if x == 0: print 99 x = 17 if x: print 1 else: print 2 f() print 17</pre>	<pre>def f(): return 7*2 def g(): print 15 x = 12 def h(): return x print h() return 2*x print f() print g()</pre>	<pre>def f(): x = 0 if x == 0: print 99 x = 17 if x: print 1 return else: print 2 print 101 f() print 17</pre>
(a) Basic Scope	(b) Nested Functions	(c) Return Statement

Figure 1: Examples of Some Interesting Python Test Cases.