

Robert Sato

rssato

CSE 210A - Programming Languages

February 5, 2021

Homework 4: WHILE Small Step Interpreter

```
~/Desktop/Graduate School/Winter 2021/CSE 210A - Programming Languages/cse210A-asgtest-master/cse210A-asgtest-hw4-
(base) robertsato@eduroam-169-233-201-101 cse210A-asgtest-hw4-whiless % ./test.sh
rm -rf while-ss
rm -rf *.pyc
cp asgn4.py while-ss
chmod u+x while-ss
/usr/bin/readlink: illegal option -- f
usage: readlink [-n] [file ...]

✓ custom-1
✓ custom-2
✓ custom-3
✓ custom-4
✓ custom-5
✓ easy-1
✓ easy-2
✓ easy-3
✓ easy-4
✓ easy-5
✓ easy-6
✓ easy-7
✓ easy-8
✓ easy-9
✓ easy-10
✓ easy-12
✓ easy-13
✓ easy-14
✓ easy-15
✓ easy-16
✓ easy-17
✓ hard-1
✓ hard-2
✓ hard-3
✓ hard-4
✓ hard-5
✓ hard-6
✓ hard-7
✗ hard-8
(from function 'check' in file tests/harness.bash, line 6,
in test file tests/hard.bats, line 39)
'check 'if ( true ) then x := 1 else z := 2' 'x := 1, {}' failed
if ( true ) then x := 1 else z := 2 == x := 1, {}
= skip, (y := 0), your code outputs Error: is_bool never set. No operations seen
self.eat: Expected type: TRUE and got type: )
Traceback (most recent call last):
  File "/while-ss", line 680, in <module>
    node = parser.parse()
  File "/while-ss", line 540, in parse
    return self.c_expr()
  File "/while-ss", line 540, in c_expr
    node = self.get_command()
  File "/while-ss", line 517, in get_command
    cond = self.b_expr()
  File "/while-ss", line 437, in b_expr
    node = self.b_term()
  File "/while-ss", line 449, in b_term
    node = self.b_factor()
  File "/while-ss", line 481, in b_factor
    node = self.expr()
  File "/while-ss", line 417, in expr
    node = self.term()
  File "/while-ss", line 400, in term
    node = self.factor()
  File "/while-ss", line 383, in factor
    self.eat(RPAREN)
  File "/while-ss", line 383, in self.eat
    raise Exception('I
Exception: Invalid syn

✗ medium-7
self.eat(RPAREN)
(from function 'check' in file tests/harness.bash, line 6,
in test file tests/medium.bats, line 1624)
'check 'while false do x := 1 ; if true then y := 1 else z := 1' 'x := skip; if
while false do x := 1 ; if true then y := 1 else z := 1 == skip; if true then
= if true then ( y := 1 ) else ( z := 1 ), {}
= y := 1, {}
= skip, (y := 1), your code outputs = skip, {}
✗ medium-8
(from function 'check' in file tests/harness.bash, line 6,
in test file tests/medium.bats, line 1631)
'check 'while false do x := 1 ; y := 1' 'x := 1, {}' failed
while false do x := 1 ; y := 1 == skip; y := 1, {}
= skip, (y := 1), your code outputs = skip, {}
= y := 1, {}
✓ medium-9
✓ medium-10
51 tests, 4 failures
✗ medium-8
(from function 'check' (base) robertsato@eduroam-169-233-201-101 cse210A-asgtest-hw4-whiless %
```

My implementation builds heavily from homework 2, the WHILE interpreter. The tokenizer and parser were virtually unchanged. The main changes involved logic for determining if a parenthesis belonged to a boolean or arithmetic expression, fixing issues with pre-existing parser logic, formatting of output, and interpreting the AST. Because this is a small step interpreter, we can no longer recursively visit tree nodes. Instead, I iteratively loop through one step of the interpreter at a time. The interpreter

returns the corresponding new state and store at each iteration. The same <https://ruslanspivak.com/lsbasi-part7/> parser code was used for this implementation.