✓

Prove total correctness

2

declaration	double answer = 1, d = 0.00001
	while (answer * answer <= num) {
adds d	answer += d;
	}
	if (answer * answer > num) {
subtracts d	answer -= d;
	}
	// {answer = sqrt(num)}

double val ← double {k=1|

loop invariant

answer ← num { 1 ≤ k <i ~~[illegible]~~ += d }

true when < num

answer ← ~~max~~ answer {0 < k < ~~[illegible]~~ +d answer−d }

answer ± d = answer = sqrt(num)

Basis property)

loop invalid

✓ answer 1·1 = 1   1−0.000001 ≈ 1 = sqrt(num)

~~Inductive property~~

Assum

At some point since answer ~~grows each iteration~~ may it will also equal num so terminates loop and returns sqrt(1) ≈ 1

~~After loop is done~~

2 continued on other page

2 continued

Prove total correctness
since code has a loop makes it harder
So invariant

answer = (answer - d) AND answer > 0

So it can terminate

satisfies criteria for loop correctness

P => I
(I and B) S {I}
(I and (not B)) => Q
loop terminated

Satisfies post condition of being sqrt (num)

it also follows the format of

{P} while B do S end {Q}

Weakest P (10)

4

1) If (x>y) y = 2x+1 else y = 3x-1 {y>3}

When is+ rose x>y y = 2x+1 and y > 3, x > 1

if not y = 3x - 1

y > 3 , x > 2

↳ W.P. {x > 1 or x > 2}

2) x = y + 10 {y > 10}

when y = 10 x = 10+10 = 20 > 20

W.P {x > 20}

3) y = y+1 ; z = x + y ; {x < z}

if last condition

z = x + y and when y > 1 → x < z

W.P {y > 1}

4) x = 3*y+1 ; y = x - 4 ; {y < 0}

3 * (x-4) + 1

3x - 12 + 1 = 3x - 11

this case y is negative so < 0

Answer logic

So x - y < 0

3x - 12 + 1 = 11/3

W.P. {y < 11/3}

So y < 0

W.P {y < 1}

If

y < 0 y = (3y+1) - 4 → 3y - 3 = 3y = 3 y = 1

5 wp for (x > 0) then y = y+2 else y = y + 3 {y>0}

$y = y + 2$ $\quad\quad$ $y = y + 3$

-2 $\quad\quad$ -3

w.p $\{y \geq -2$ or $y \geq -3\}$

5

Java do while

do {
code to be executed }
while (condition)

so if → loop: code to be executed
if (condition) = false goto out
goto loop
out: ...

Ada for

for I in start -> complete loop
I = first
loop: If I < last go out
→ So I = I + 1
go to loop
out: ...

C++ if then else

if (condition) {
Statement 1 ;
Statement 2; } } → code
else {
three → statement } ;
}
→ So if condition = = false go to else
code
go to out
else:
statement }

C For

for (expressions 1; 2; 3)
eval (expression 1);
loop = control = eval (expression 2)
if control == 0
go to out
evaluate (expression 3)
goto loop
out: ...

C Switch

Switch (i) {
case 1:
Statement-1 x;
break;

case 2:
statementsy;
break;
default }

↳ if matched with case 1 then i=1
if case 2 then i = 2
if none
↳ goto default

6) Denotational Semantics

→ taken from Dynamic semantics powerpoint

Python for

M (For f, state state) = Mf (f, M(f.initial, state)

Mf (For f, state state) =

{m (f, Mf (f.pos-1, M (f.body, state)))

if (M (f.test, state) is true)

{state otherwise}

Java class

Add simple varmap

Mel) = VARMAP (i, S)

then S = { <i1, v1> <i2, v2> ... }

C++ function

VarmaD

if M3 (B, S)

then

S

C++ ~~function~~ switch
when switch := true;

~~[redacted]~~

m_1 (while B do L, s) Δ =
if m_b (B, s) == error
then error
else if m_b (B, s)
== false
then s
else if m_{s_1} (L, s) == error
then error
else m_1 (while B do L,
m_{s_1} (L, s)).

then a switch (not switch) in case where stop

end loop

Java boolean expression

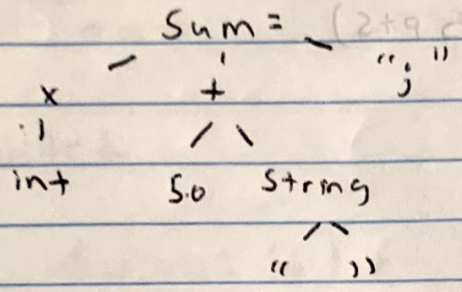
m_b (B, s) if VARMAP (I, s) = undef
for some I in B
then error
else
B
VarMap

Decorated tree

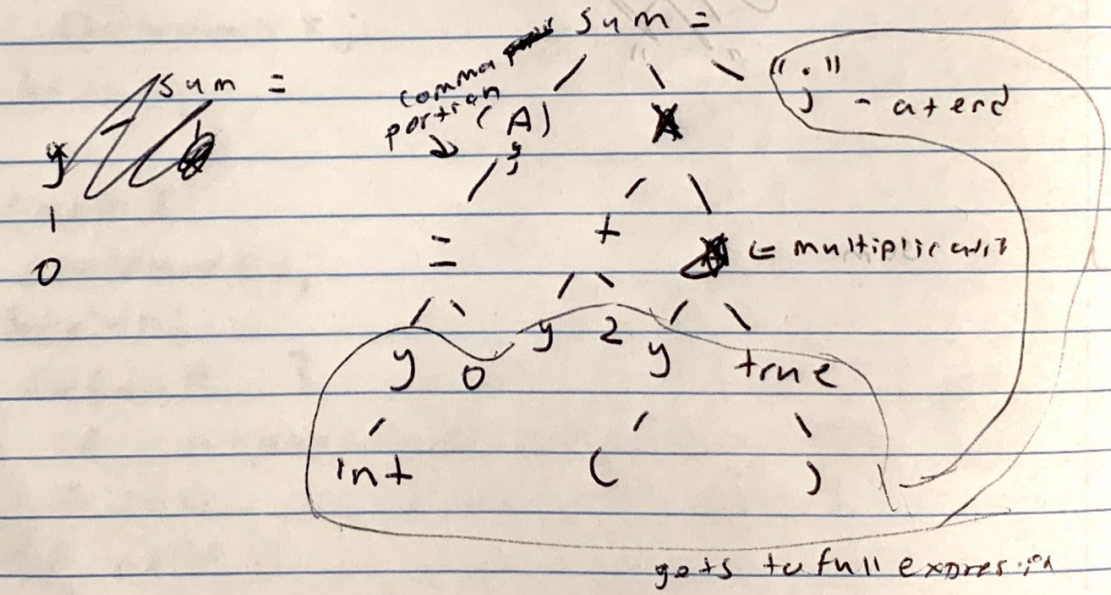
1)

decorated tree (15)

a)



b)



c)