\

][=-It was filled in between 17th April and 1st May 2018  
Thanks to whoever did it

the date is 29th March 2018

i have opened this document and it is empty

pls halp

the date is 17th April 2018

it’s still empty

1. a) ∀x:a[P(leaf x)] ^ ∀t1,t2:Tree a[P(t1) ^ P(t2) → P(Node t1 t2)] → ∀t:Tree a[P(t)]

b) ∀x:Int[P(Val x)] ^ ∀t:Term[P(t) → P(UMinus t)] ^ ∀t1,t2:Term[P(t1) ^ P(t2) → P(Mult t1 t2)]

→ ∀t:Term[P(t)]

c)

i) a) -6

b) Mult (Val -3) (Val 2)

c) True

d) -6

c) -6

ii) Base Case:

Take ∀i:Int(Val i) : Term arbitrary.

(i.e. take just an arbitrary i:Int)

To show: eval(Val i) = wSign (eval (rip (Val i))) (pos (Val i))

1. wSign (eval (rip (Val i))) (pos (Val i))
2. = wSign (eval (rip (val i))) True (By def pos)
3. = (eval (rip (val i))) (By def wSign)
4. = eval (val i) (By def rip)

Inductive step:

Take t:Term arbitrary

IH: eval(t) = wSign (eval (rip t)) (pos t)

To show: eval(Uminus t) = wSign(eval(rip(Uminus t)) (pos (UMinus t))

1. wSign (eval( rip (Uminus t))) (pos (Uminus t))
2. wSign (eval (rip t)) (not (pos t)) (By def rip & def pos)
3. - wSign (eval (rip t)) (not (pos t)) (By lemma B)
4. - eval (t) (By IH)
5. Eval (Uminus t) (By def eval)

Take t1,t2:Term arbitrary.

IH: eval(t1) = wSign (eval (rip (t1)) (pos (t1))

eval(t2) = wSign (eval (rip (t2)) (pos (t2))

To show: eval(Mult t1 t2) = wSign (eval (rip (Mult t1 t2)) (pos (Mult t1 t2))

1. wSign (eval (rip (Mult t1 t2)) (pos (Mult t1 t2))
2. = wSign (eval (rip (Mult t1 t2)) (pos t1 == pos t2) (By def pos)
3. = wSign (eval (Mult (rip t1) (rip t2)) (pos t1 == pos t2) (By def rip)
4. = wSign (eval (rip t1) \* eval (rip t2)) (pos t1 == pos t2) (By def eval)
5. = (wSign (eval (rip t1)) (pos t1)) \* (wSign (eval (rip t2)) (pos t2)) (By C)
6. = eval (t1) \* eval (t2) (By IH)
7. = eval (Mult t1 t2) (By def eval)

2.

a)

[4, 5, 5, 2] - notice the type here

b)

i) An exception will be thrown when b is an empty array

ii) At line 15

iii) An ArrayIndexOutOfBoundsException will be thrown due to the fact that there is no element in index 0

iv) a ≠ null and a.length > 0

c)

i) 0 <= i <= a.length - 1

ii) [i = 0 -> prv = 0] and [i > 0 -> prv = a[i - 1]]

iii) k [1 .. i ). a [k] = (a0[k - 1] + a0[k]+ a0[k + 1]) / 3

iv) k [i .. a.length ). a [k] = a0[k]

d)

*Prove all the array access within the loop is valid*

Given:

1. 0 <= i <= a.length - 1 INV
2. i < a.length - 1 COND

To show:

alpha) 0 <= i < a.length - 1

beta) 0 <= i+1 <= a.length - 1

Proof:

alpha follows by 1

3) 0 <=i < a.length - 1 by 1 and 2

beta follows by 3 and arithmetic

*Prove array access in line 15 is valid*

Given:

1. 0 <= i <= a.length - 1 INV
2. i >= a.length - 1 ~COND

To show:

alpha) 0 <= i <= a.length - 1

Proof:

3) i = a.length - 1

alpha follows by 3

Alternatively, perhaps the MID on line 14 can be used

e) It would all tend to 0