

NaturalSemantics.pdf



Juanma21_



Teoría de Los Lenguajes de Programación



4º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga



MÁSTER EN

**Inteligencia Artificial
& Data Management**

MADRID

Formamos
talento para un futuro
Sostenible

saber más



Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45

NaturalSemantics.hs

```
1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  --
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

Consulta condiciones aquí



do your thing

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

WUOLAH

1/4

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   Si es True                                         s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos Holandés con una garantía de hasta 100.000 euros por depositante. Consulta más información en [ing.es](#)

Que te den 10 € para gastar
es una fantasía.
ING lo hace realidad.

Abre la **Cuenta NoCuenta** con el código
[WUOLAH10](#), haz tu primer pago y llévate 10 €.

Quiero el cash

[Consulta condiciones aquí](#)



do your thing

Teoría de Los Lenguajes de P...



Comparte estos flyers en tu clase y consigue más dinero y recompensas



- 1 Imprime esta hoja
- 2 Recorta por la mitad
- 3 Coloca en un lugar visible para que tus compañeros puedan escanear y acceder a apuntes
- 4 Llévate dinero por cada descarga de los documentos descargados a través de tu QR

Banco de apuntes de la

WUOLAH



```

104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y")))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y")) (Add (Mult (N 3) (N 2)) (N 5)))
147 -- exp3 = (8 * y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y")) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156

```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en [ing.es](#).

13/1/25, 13:45

NaturalSemantics.hs

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```



Consulta condiciones aquí



do your thing

WUOLAH

NaturalSemantics.hs

```

1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  -- -----
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene una cobertura máxima de hasta 100.000 euros por depositante. Consulta más información en inglés.

13/1/25, 13:45 NaturalSemantics.hs

```
104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y"))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y"))) (Add (Mult (N 3) (N 2)) (N 5))
147 -- exp3 = (8* y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y"))) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156
```

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

3/4

Consulta condiciones aquí



do your thing

WUOLAH

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```

NaturalSemantics.hs

```

1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  -- -----
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stmt State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45 NaturalSemantics.hs

```
52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57   -}
58
59 -- Sentencia Swap
60 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
61   where
62     Final s1 = nsStm (Inter (Ass x (V y)) s)
63
64 {-
65
66 Si es True
67           <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68 s''''                                     <For s1 b s2 s3, s> -> s'''''
69   [for ns] -----
70   ----- if B[b]s = True
71
72
73 Si es False
74           <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76           <For s1 b s2 s3, s> -> s'
77
78 -}
79
80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s'''''
83   where
84     Final s'    = nsStm (Inter s1 s)
85     Final s''   = nsStm (Inter s3 s')
86     Final s''' = nsStm (Inter s2 s'')
87     Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
88
89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
91   where
92     Final s'    = nsStm (Inter s1 s)
93
94
95 reduce :: Aexp -> Aexp
96
97 -- Entero
98 reduce (N n) = (N n)
99
100 -- Variable
101 reduce (V x) = (V x)
102
103 -- Suma 1
```

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

2/4

Consulta condiciones aquí



do your thing

WUOLAH

```

104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y")))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y")) (Add (Mult (N 3) (N 2)) (N 5)))
147 -- exp3 = (8 * y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y")) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156

```

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45

NaturalSemantics.hs

```
1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  --
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

Consulta condiciones aquí



do your thing

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

WUOLAH

1/4

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   Si es True                                         s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

```

104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y")))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y")) (Add (Mult (N 3) (N 2)) (N 5)))
147 -- exp3 = (8 * y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y")) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156

```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en [ing.es](#).

13/1/25, 13:45

NaturalSemantics.hs

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```



Consulta condiciones aquí



do your thing

WUOLAH

NaturalSemantics.hs

```

1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  -- -----
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stmt State  -- <s, s>
18     | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45 NaturalSemantics.hs

```
104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y"))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y"))) (Add (Mult (N 3) (N 2)) (N 5))
147 -- exp3 = (8* y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y"))) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156
```

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

3/4

Consulta condiciones aquí



do your thing

WUOLAH

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```

NaturalSemantics.hs

```

1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  -- -----
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

```
13/1/25, 13:45                                         NaturalSemantics.hs

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57   -}
58
59 -- Sentencia Swap
60 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
61   where
62     Final s1 = nsStm (Inter (Ass x (V y)) s)
63
64 {-
65
66 Si es True
67           <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68 s''''                                     <For s1 b s2 s3, s> -> s'''''
69   [for ns] -----
70   ----- if B[b]s = True
71
72
73 Si es False
74           <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76           <For s1 b s2 s3, s> -> s'
77
78 -}
79
80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s'''''
83   where
84     Final s'      = nsStm (Inter s1 s)
85     Final s''     = nsStm (Inter s3 s')
86     Final s'''    = nsStm (Inter s2 s'')
87     Final s''''' = nsStm (Inter (For Skip b s2 s3) s''''')
88
89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
91   where
92     Final s'      = nsStm (Inter s1 s)
93
94
95 reduce :: Aexp -> Aexp
96
97 -- Entero
98 reduce (N n) = (N n)
99
100 -- Variable
101 reduce (V x) = (V x)
102
103 -- Suma 1
```

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

2/4

Consulta condiciones aquí



do your thing

WUOLAH

```

104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y")))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y")) (Add (Mult (N 3) (N 2)) (N 5)))
147 -- exp3 = (8 * y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y")) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156

```

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45

NaturalSemantics.hs

```
1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  --
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

Consulta condiciones aquí



do your thing

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

WUOLAH

1/4

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   Si es True                                         s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

```

104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y")))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y")) (Add (Mult (N 3) (N 2)) (N 5)))
147 -- exp3 = (8 * y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y")) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156

```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en [ing.es](#).

13/1/25, 13:45

NaturalSemantics.hs

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```



Consulta condiciones aquí



do your thing

WUOLAH

NaturalSemantics.hs

```

1  -- -----
2  -- Natural Semantics for WHILE 2020.
3  -- Examen de Lenguajes de Programación. UMA.
4  --
5  -- Apellidos, Nombre: Cardeñosa Borrego, Juan Manuel
6  -- -----
7
8  module NaturalSemantics where
9
10 import           While20
11
12 updateState :: State -> Var -> Z -> State
13 updateState s x v y = if x == y then v else s y
14
15 -- representation of configurations for While
16
17 data Config = Inter Stm State  -- <s, s>
18           | Final State      -- s
19
20 -- representation of the transition relation <s, s> -> s'
21
22 nsStm :: Config -> Config
23
24 -- x := a
25
26 nsStm (Inter (Ass x a) s) = Final (updateState s x (aVal a s))
27
28 -- skip
29
30 nsStm (Inter Skip s) = Final s
31
32 -- s1; s2
33
34 nsStm (Inter (Comp ss1 ss2) s) = Final s''
35   where
36     Final s' = nsStm (Inter ss1 s)
37     Final s'' = nsStm (Inter ss2 s')
38
39 -- if b then s1 else s2
40
41 nsStm (Inter (If b ss1 ss2) s)
42   | bVal b s = Final s'
43   where
44     Final s' = nsStm (Inter ss1 s)
45
46 nsStm (Inter (If b ss1 ss2) s)
47   | not(bVal b s) = Final s'
48   where
49     Final s' = nsStm (Inter ss2 s)
50
51 -----
```

```

52 -- NO MODIFICAR EL CODIGO DE ARRIBA
53 -----
54
55 {-                                     <x := y, s> -> s'
56   [swap ns] ----- <Swap x y, s> -> s'[y -> A[x]s]
57 -}
58
59
60
61 -- Sentencia Swap
62 nsStm (Inter (Swap x y) s) = Final (updateState s1 y (s x))
63   where
64     Final s1 = nsStm (Inter (Ass x (V y)) s)
65
66 {-                                     Si es True
67   <s1, s> -> s' <s3, s'> -> s'' <s2, s''> -> s''' <For Skip b s2 s3, s'''> ->
68   Si es True                                         s'''
69   [for ns] ----- if B[b]s = True
70   ----- if B[b]s = True
71   <For s1 b s2 s3, s> -> s''''
```

72

```

73 Si es False
74   <s1,s> -> s'
75   [for ns] ----- if B[b]s = False
76   <For s1 b s2 s3, s> -> s'
```

77

```

78 -}
```

79

```

80 -- Sentencia for
81 -- Si es true
82 nsStm(Inter (For s1 b s2 s3) s) | bVal b s = Final s''''
```

83 where

```

84   Final s'    = nsStm (Inter s1 s)
85   Final s''   = nsStm (Inter s3 s')
86   Final s''' = nsStm (Inter s2 s'')
87   Final s'''' = nsStm (Inter (For Skip b s2 s3) s''')
```

88

```

89 -- Si es false
90 nsStm(Inter (For s1 b s2 s3) s) | not(bVal b s) = Final s'
```

91 where

```

92   Final s'    = nsStm (Inter s1 s)
```

93

94

```

95 reduce :: Aexp -> Aexp
```

96

```

97 -- Entero
98 reduce (N n) = (N n)
```

99

```

100 -- Variable
101 reduce (V x) = (V x)
```

102

```

103 -- Suma 1
```

Esto no son apuntes pero **tiene un 10 asegurado** (y lo vas a disfrutar igual).

Abre la Cuenta NoCuenta con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Me interesa



1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos y tiene como una garantía de hasta 100.000 euros por depositista. Consulta más información en inglés.

13/1/25, 13:45 NaturalSemantics.hs

```
104 reduce (Add (N n1) (N n2)) = reduce (N (n1 + n2))
105
106 -- Suma 2
107 reduce (Add a1 a2) = Add (reduce a1') (reduce a2')
108   where
109     a1' = reduce a1
110     a2' = reduce a2
111
112
113 -- Resta 1
114 reduce (Sub (N n1) (N n2)) = reduce (N (n1 - n2))
115
116 -- Resta 2
117 reduce (Sub a1 a2) = Sub (reduce a1') (reduce a2')
118   where
119     a1' = reduce a1
120     a2' = reduce a2
121
122
123 -- Mult 1
124 reduce (Mult (N n1) (N n2)) = reduce (N (n1 * n2))
125
126 -- Mult 2
127 reduce (Mult a1 a2) = Mult (reduce a1') (reduce a2')
128   where
129     a1' = reduce a1
130     a2' = reduce a2
131
132
133 exp1 :: Aexp
134 exp1 = Mult (Add (Add (N 3) (N 6)) (V "x")) (Sub (Mult (N 4) (N 8)) (V "y"))
135 -- exp1 = ((3 + 6) + (x)) * ((4 * 8) - y)
136 -- reduce exp1
137 -- Solucion: (9 + x) * (32 - y)
138
139 exp2 :: Aexp
140 exp2 = Add (V "x") (Mult (N 5) (N 3))
141 -- exp2 = x + (5 * 3)
142 -- reduce exp2
143 -- Solucion: x + 15
144
145 exp3 :: Aexp
146 exp3 = Add (Mult (N 8) (V "y"))) (Add (Mult (N 3) (N 2)) (N 5))
147 -- exp3 = (8* y) + ((3 + 2) * 5)
148 -- reduce exp3
149 -- Solucion: (8 * y) + 11
150
151 exp4 :: Aexp
152 exp4 = Add (Mult (N 8) (V "y"))) (Add (Mult (Mult (N 4) (Mult (N 2) (Mult (N 3) (N 4)))))) (N 2)) (N 5))
153 -- exp4 = (8 * y) + ((4 * (2 * (3 * 2))) + 5)
154 -- reduce exp4
155 -- Solucion: (8 * y) + 197
156
```

localhost:50364/372ba9cd-c46f-4de7-8bbe-7dd5a1256d0b/

3/4

Consulta condiciones aquí



do your thing

WUOLAH

```
157  
158  
159  
160  
161  
162 -- Swap como funcion para probar, se pide el stm (arriba está hecho)  
163 s1 :: State  
164 s1 "x" = 3  
165 s1 "y" = 4  
166  
167 exp5 :: Aexp  
168 exp5 = Add (Mult (V "x") (N 5) ) (V "y")  
169  
170 -- aVal exp5 s1  
171 -- Da 19  
172 -- aVal exp5 (swap s1 "x" "y")  
173 -- Da 23  
174  
175
```