



COMP4161: Advanced Topics in Software Verification

INV

Gerwin Klein, June Andronick, Christine Rizkallah, Miki Tanaka
S2/2018

data61.csiro.au



Last Time



- Weakest preconditions
- Verification conditions
- Example program proofs
- Arrays, pointers

Content



- Intro & motivation, getting started [1]

- Foundations & Principles
 - Lambda Calculus, natural deduction [1,2]
 - Higher Order Logic [3^a]
 - Term rewriting [4]

- Proof & Specification Techniques
 - Inductively defined sets, rule induction [5]
 - Datatypes, recursion, induction [6, 7]
 - Hoare logic, proofs about programs, invariants [8^b,9]
 - (mid-semester break)
 - C verification [10]
 - CakeML, Isar [11^c]
 - Concurrency [12]

^aa1 due; ^ba2 due; ^ca3 due

Today



Practice with invariants!

Recall:

- it needs to be an invariant
- it needs to be enough

Example 1



```
A := 0;
```

```
B := 0;
```

```
WHILE  $A \neq a$ 
```

```
DO
```

```
     $B := B + b$ ;
```

```
     $A := A + 1$ 
```

```
OD
```

Example 1



```
A := 0;  
B := 0;  
  
WHILE A ≠ a  
DO  
    B := B + b;  
    A := A + 1  
OD  
{ B = b * a }
```

Example 1



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

WHILE $A \neq a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

Example 1



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

$INV \{ B = b * A \}$

WHILE $A \neq a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

Example 1



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

$\text{INV } \{ B = b * A \}$

$\text{WHILE } A \neq a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

$$0 = b * 0$$

$$B = b * A \wedge A \neq a \longrightarrow B + b = b * (A + 1)$$

$$B = b * A \wedge A = a \longrightarrow B = b * a$$

Example 2



$A := 0;$

$B := 0;$

WHILE $A < a$

DO

$B := B + b;$

$A := A + 1$

OD

Example 2



$A := 0;$

$B := 0;$

WHILE $A < a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

Example 2



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

WHILE $A < a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

Example 2



```
{ a ≥ 0 ∧ b ≥ 0 }  
A := 0;  
B := 0;  
INV { B = b * A }  
WHILE A < a  
DO  
    B := B + b;  
    A := A + 1  
OD  
{ B = b * a }
```

Example 2



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

$\text{INV } \{ B = b * A \}$

$\text{WHILE } A < a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

$$0 = b * 0$$

$$B = b * A \wedge A < a \longrightarrow B + b = b * (A + 1)$$

$$B = b * A \wedge A \geq a \longrightarrow B = b * a$$

Example 2



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := 0;$

$B := 0;$

$\text{INV } \{ B = b * A \} \wedge A \leq a$

WHILE $A < a$

DO

$B := B + b;$

$A := A + 1$

OD

$\{ B = b * a \}$

$0 = b * 0 \wedge 0 \leq a$

$B = b * A \wedge A < a \longrightarrow B + b = b * (A + 1)$
 $\wedge A \leq a \qquad \wedge A + 1 \leq a$

$B = b * A \wedge A \geq a \longrightarrow B = b * a$
 $\wedge A \leq a$

Example 3



$A := a;$

$B := 1;$

WHILE $A \neq 0$

DO

$B := B * b;$

$A := A - 1$

OD

Example 3



$A := a;$

$B := 1;$

WHILE $A \neq 0$

DO

$B := B * b;$

$A := A - 1$

OD

$\{ B = b^a \}$

Example 3



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := a;$

$B := 1;$

WHILE $A \neq 0$

DO

$B := B * b;$

$A := A - 1$

OD

$\{ B = b^a \}$

Example 3



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := a;$

$B := 1;$

$INV \{ B = b^{a-A} \}$

WHILE $A \neq 0$

DO

$B := B * b;$

$A := A - 1$

OD

$\{ B = b^a \}$

Example 3



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := a;$

$B := 1;$

$\text{INV } \{ B = b^{a-A} \}$

WHILE $A \neq 0$

DO

$B := B * b;$

$A := A - 1$

OD

$\{ B = b^a \}$

$$1 = b^{a-a}$$

$$B = b^{a-A} \wedge A \neq 0 \longrightarrow B * b = b^{a-(A-1)}$$

$$B = b^{a-A} \wedge A = 0 \longrightarrow B = b^a$$

Example 4



$X := x;$

$Y := [];$

WHILE $X \neq []$

DO

$Y := (hd\ X \# Y);$

$X := tl\ X$

OD

Example 4



$X := x;$

$Y := [];$

WHILE $X \neq []$

DO

$Y := (hd\ X \# Y);$

$X := tl\ X$

OD

$\{ Y = rev\ x \}$

Example 4



{ *True* }

$X := x;$

$Y := [];$

WHILE $X \neq []$

DO

$Y := (hd\ X \# Y);$

$X := tl\ X$

OD

{ $Y = rev\ x$ }

Example 4



```
{ True }  
X := x;  
Y := [];  
INV { (rev X)@Y = rev x }  
WHILE X ≠ []  
  
DO  
  Y := (hd X#Y);  
  X := tl X  
OD  
{ Y = rev x }
```


Example 4



```
{ True }
X := x;
Y := [];      (rev x)@[] = rev x
INV { (rev X)@Y = rev x }
WHILE X ≠ []
    (rev X)@Y = rev x ∧ X ≠ [] →
    (rev (tl X))@((hd X)#Y) = rev x
DO
    Y := (hd X#Y);
    X := tl X
OD      (rev X)@Y = rev x ∧ X = [] → Y = rev x
{ Y = rev x }
```

Example 5



```
A := a; B := b; C := 1;
```

```
WHILE B ≠ 0  
DO
```

```
    WHILE (B mod 2 = 0)
```

```
        DO
```

```
            A := A * A;
```

```
            B := B div 2;
```

```
        OD
```

```
        C := C * A;
```

```
        B := B - 1
```

```
    OD
```

Example 5

Try with $b = 10 = 2^1 + 2^3$ or $b = 12 = 2^2 + 2^3$ (and e.g. $a=3$)



```
A := a; B := b; C := 1;
```

```
WHILE B ≠ 0  
DO
```

```
    WHILE (B mod 2 = 0)
```

```
        DO
```

```
            A := A * A;
```

```
            B := B div 2;
```

```
        OD
```

```
        C := C * A;
```

```
        B := B - 1
```

```
    OD
```

Example 5

Try with $b = 10 = 2^1 + 2^3$ or $b = 12 = 2^2 + 2^3$ (and e.g. $a=3$)



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := a; B := b; C := 1;$

WHILE $B \neq 0$

DO

 WHILE $(B \bmod 2 = 0)$

 DO

$A := A * A;$

$B := B \text{ div } 2;$

 OD

$C := C * A;$

$B := B - 1$

OD

$\{ C = a^b \}$

Example 5

Try with $b = 10 = 2^1 + 2^3$ or $b = 12 = 2^2 + 2^3$ (and e.g. $a=3$)



```
{  $a \geq 0 \wedge b \geq 0$  }  
A := a; B := b; C := 1;  
INV {  $a^b = C * A^B$  }  
WHILE B  $\neq$  0  
DO  
  INV {  $a^b = C * A^B$  }  
    WHILE (B mod 2 = 0)  
  
      DO  
        A := A * A;  
        B := B div 2;  
      OD  
      C := C * A;  
      B := B - 1  
    OD  
  {  $C = a^b$  }
```

Example 5

Try with $b = 10 = 2^1 + 2^3$ or $b = 12 = 2^2 + 2^3$ (and e.g. $a=3$)



$\{ a \geq 0 \wedge b \geq 0 \}$

$A := a; B := b; C := 1;$

$$a^b = 1 * a^b$$

INV $\{ a^b = C * A^B \}$

WHILE $B \neq 0$

$$a^b = C * A^B \wedge B \neq 0 \longrightarrow a^b = (C * A) * A^{B-1}$$

DO

INV $\{ a^b = C * A^B \}$

WHILE $(B \bmod 2 = 0)$

$$a^b = C * A^B \wedge B \bmod 2 = 0 \longrightarrow a^b = C * (A * A)^{B \text{ div } 2}$$

DO

$A := A * A;$

$B := B \text{ div } 2;$

OD

$C := C * A;$

$B := B - 1$

OD

$$a^b = C * A^B \wedge B = 0 \longrightarrow C = a^b$$

$\{ C = a^b \}$

Example 6



$l := 0; u := \text{length } A - 1;$

WHILE $l \leq u$
DO

 WHILE $l < \text{length } A \wedge A[l] \leq \text{piv}$ DO $l := l + 1$ OD;

 WHILE $0 < u \wedge \text{piv} \leq A[u]$ DO $u := u - 1$ OD;

 IF $l \leq u$ THEN $A := A[l := A[u], u := A[l]]$ ELSE SKIP FI
OD

Example 6



$l := 0; u := \text{length } A - 1;$

WHILE $l \leq u$
DO

 WHILE $l < \text{length } A \wedge A[l] \leq \text{piv}$ DO $l := l + 1$ OD;

 WHILE $0 < u \wedge \text{piv} \leq A[u]$ DO $u := u - 1$ OD;

 IF $l \leq u$ THEN $A := A[l := A[u], u := A[l]]$ ELSE SKIP FI
OD

$\{ \text{LEQ } A \ u \wedge \text{EQ } u \ l \wedge \text{GEP } A \ l \}$

Example 6



$LEQ\ A\ n = \forall k. k < n \longrightarrow A!k \leq piv$

$QEQ\ A\ n = \forall k. n < k < length\ A \longrightarrow A!k \geq piv$

$EQ\ A\ n\ m = \forall k. n < k < m \longrightarrow A!k = piv$

$\{ 0 < length\ A \}$

$l := 0; u := length\ A - 1;$

WHILE $l \leq u$

DO

WHILE $l < length\ A \wedge A!l \leq piv$ DO $l := l + 1$ OD;

WHILE $0 < u \wedge piv \leq A!u$ DO $u := u - 1$ OD;

IF $l \leq u$ THEN $A := A[l := A!u, u := A!l]$ ELSE SKIP FI
OD

$\{ LEQ\ A\ u \wedge EQ\ u\ l \wedge GEP\ A\ l \}$

Example 6



$LEQ\ A\ n = \forall k. k < n \longrightarrow A!k \leq piv$

$QEQ\ A\ n = \forall k. n < k < length\ A \longrightarrow A!k \geq piv$

$EQ\ A\ n\ m = \forall k. n < k < m \longrightarrow A!k = piv$

$\{ 0 < length\ A \}$

$l := 0; u := length\ A - 1;$

$INV\ \{ LEQ\ A\ l \wedge GEP\ A\ u \wedge u < length\ A \wedge l \leq length\ A \}$

WHILE $l \leq u$

DO

$INV\ \{ LEQ\ A\ l \wedge GEP\ A\ u \wedge u < length\ A \wedge l \leq length\ A \}$

WHILE $l < length\ A \wedge A!l \leq piv$ DO $l := l + 1$ OD;

$INV\ \{ LEQ\ A\ l \wedge GEP\ A\ u \wedge u < length\ A \wedge l \leq length\ A \}$

WHILE $0 < u \wedge piv \leq A!u$ DO $u := u - 1$ OD;

IF $l \leq u$ THEN $A := A[l := A!u, u := A!l]$ ELSE SKIP FI

OD

$\{ LEQ\ A\ u \wedge EQ\ u\ l \wedge GEP\ A\ l \}$

Example 7

Reminder:

datatype ref = Ref int | Null

Pointer access: $p \rightarrow \text{field}$

Pointer update: $p \rightarrow \text{field} ::= v$

Definition:

"*List next p Ps*" is a linked list, starting at pointer p following the next pointer through the function *next*, and where Ps contains the list of the pointers of the linked list.

$\{ \text{List next } p \ Ps \wedge X \in Ps \}$

WHILE $p \neq \text{Null} \wedge p \neq \text{Ref } X$

DO

$p := p \rightarrow \text{next};$

OD



Example 7

Reminder:

datatype ref = Ref int | Null

Pointer access: $p \rightarrow \text{field}$

Pointer update: $p \rightarrow \text{field} ::= v$

Definition:

"*List* *nxt* *p* *Ps*" is a linked list, starting at pointer *p* following the next pointer through the function *nxt*, and where *Ps* contains the list of the pointers of the linked list.

$\{ \text{List } \text{nxt } p \ Ps \wedge X \in Ps \}$

WHILE $p \neq \text{Null} \wedge p \neq \text{Ref } X$

DO

$p := p \rightarrow \text{nxt};$

OD

$\{ p = \text{Ref } X \}$



Example 7

Reminder:

datatype ref = Ref int | Null

Pointer access: $p \rightarrow \text{field}$

Pointer update: $p \rightarrow \text{field} ::= v$

Definition:

"*List* *nxt* *p* *Ps*" is a linked list, starting at pointer *p* following the next pointer through the function *nxt*, and where *Ps* contains the list of the pointers of the linked list.

$\{ \text{List } \text{nxt } p \ Ps \wedge X \in Ps \}$

WHILE $p \neq \text{Null} \wedge p \neq \text{Ref } X$

DO

$p := p \rightarrow \text{nxt};$

OD

$\{ p = \text{Ref } X \}$



Example 7

Reminder:

datatype ref = Ref int | Null

Pointer access: $p \rightarrow \text{field}$

Pointer update: $p \rightarrow \text{field} ::= v$

Definition:

"*List* nxt p P_s " is a linked list, starting at pointer p following the next pointer through the function nxt , and where P_s contains the list of the pointers of the linked list.

$\{ \text{List } nxt \ p \ P_s \wedge X \in P_s \}$

INV $\{ \exists Q_s. \text{List } nxt \ p \ Q_s \wedge X \in Q_s \}$

WHILE $p \neq \text{Null} \wedge p \neq \text{Ref } X$

DO

$p := p \rightarrow \text{next};$

OD

$\{ p = \text{Ref } X \}$



Example 7

Reminder:

datatype ref = Ref int | Null

Pointer access: $p \rightarrow \text{field}$

Pointer update: $p \rightarrow \text{field} ::= v$

Definition:

"List $\text{next } p \text{ } P_s$ " is a linked list, starting at pointer p following the next pointer through the function next , and where P_s contains the list of the pointers of the linked list.



$\{ \text{List next } p \text{ } P_s \wedge X \in P_s \}$

$\exists Q_s. \text{List next } p \text{ } Q_s \wedge X \in Q_s$

INV $\{ \exists Q_s. \text{List next } p \text{ } Q_s \wedge X \in Q_s \}$

WHILE $p \neq \text{Null} \wedge p \neq \text{Ref } X$

$\exists Q_s. \text{List next } p \text{ } Q_s \wedge X \in Q_s$

$\wedge p \neq \text{Null} \wedge p \neq \text{Ref } X \rightarrow$

$\exists Q_s. \text{List next } (p \rightarrow \text{next}) \text{ } Q_s \wedge X \in Q_s$

DO

$p := p \rightarrow \text{next};$

OD

$\exists Q_s. \text{List next } p \text{ } Q_s \wedge X \in Q_s$

$\wedge (p = \text{Null} \vee p = \text{Ref } X) \rightarrow p = \text{Ref } X$

$\{ p = \text{Ref } X \}$

Example 8



What is Isabelle function doing?

fun f :: 'a list \Rightarrow ' a list \Rightarrow ' a list where

f [] ys = ys|

f xs [] = xs|

f (x#xs) (y#ys) = x#y# f xs ys

Example 8



What is the Isabelle function doing?

```
fun splice :: 'a list ⇒ 'a list ⇒ 'a list where
  splice [] ys = ys |
  splice xs [] = xs |
  splice (x#xs) (y#ys) = x#y# f xs ys
```

Example 8



What is is Isabelle function doing?

```
fun splice :: 'a list ⇒ 'a list ⇒ 'a list where
  splice [] ys = ys |
  splice xs [] = xs |
  splice (x#xs) (y#ys) = x#y# f xs ys
```

Let's write it with linked lists!

Example 8



$\{ \text{List } \text{nxt } p \ Ps \wedge \text{List } \text{nxt } q \ Qs \wedge (\text{set } Ps \cap \text{set } Qs) = \{\} \wedge \text{size } Qs \leq \text{size } Ps \}$

$\{ \text{List } \text{nxt } p \ (\text{splice } Ps \ Qs) \}$

Example 8



$\{ \text{List } \text{next } p \ Ps \wedge \text{List } \text{next } q \ Qs \wedge (\text{set } Ps \cap \text{set } Qs) = \{\} \wedge \text{size } Qs \leq \text{size } Ps \}$
 $pp := p;$

```
WHILE  $q \neq \text{Null}$ 
DO
   $qq := q \rightarrow \text{next}; q \rightarrow \text{next} := pp \rightarrow \text{next}; pp \rightarrow \text{next} = q; pp := q \rightarrow \text{next}; q := qq;$ 
OD
 $\{ \text{List } \text{next } p \ (\text{splice } Ps \ Qs) \}$ 
```

Example 8

List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*



$\{ \textit{List} \textit{nxt} \textit{p} \textit{Ps} \wedge \textit{List} \textit{nxt} \textit{q} \textit{Qs} \wedge (\textit{set} \textit{Ps} \cap \textit{set} \textit{Qs}) = \{\} \wedge \textit{size} \textit{Qs} \leq \textit{size} \textit{Ps} \}$

pp := *p*;

INV {

}

WHILE *q* ≠ *Null*

DO

qq := *q* → *nxt*; *q* → *nxt* := *pp* → *nxt*; *pp* → *nxt* = *q*; *pp* := *q* → *nxt*; *q* := *qq*;

OD

{ *List* *nxt* *p* (*splice* *Ps* *Qs*) }

Example 8

List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*



$\{ \textit{List} \textit{nxt} \textit{p} \textit{Ps} \wedge \textit{List} \textit{nxt} \textit{q} \textit{Qs} \wedge (\textit{set} \textit{Ps} \cap \textit{set} \textit{Qs}) = \{\} \wedge \textit{size} \textit{Qs} \leq \textit{size} \textit{Ps} \}$

pp := *p*;

INV { $\exists \textit{PPs}$

List *nxt* *pp* *PPs*

}

WHILE *q* \neq *Null*

DO

qq := *q* \rightarrow *nxt*; *q* \rightarrow *nxt* := *pp* \rightarrow *nxt*; *pp* \rightarrow *nxt* = *q*; *pp* := *q* \rightarrow *nxt*; *q* := *qq*;

OD

{ *List* *nxt* *p* (*splice* *Ps* *Qs*) }

Example 8



List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*

$\{ \textit{List} \textit{nxt} \textit{p} \textit{Ps} \wedge \textit{List} \textit{nxt} \textit{q} \textit{Qs} \wedge (\textit{set} \textit{Ps} \cap \textit{set} \textit{Qs}) = \{\} \wedge \textit{size} \textit{Qs} \leq \textit{size} \textit{Ps} \}$

pp := *p*;

INV { $\exists \textit{PPs} \textit{QQs}$

$\textit{List} \textit{nxt} \textit{pp} \textit{PPs} \wedge \textit{List} \textit{nxt} \textit{q} \textit{QQs}$

}

WHILE *q* ≠ *Null*

DO

$\textit{qq} := \textit{q} \rightarrow \textit{nxt}; \textit{q} \rightarrow \textit{nxt} := \textit{pp} \rightarrow \textit{nxt}; \textit{pp} \rightarrow \textit{nxt} = \textit{q}; \textit{pp} := \textit{q} \rightarrow \textit{nxt}; \textit{q} := \textit{qq};$

OD

$\{ \textit{List} \textit{nxt} \textit{p} (\textit{splice} \textit{Ps} \textit{Qs}) \}$

Example 8



List $\text{next } p \ Ps = \text{Path } \text{next } p \ Ps \ \text{Null}$

Path $\text{next } p \ Ps \ \text{Null}$ is a linked list from p to q following function *next* and containing list of pointers P_s

$\{ \text{List } \text{next } p \ Ps \wedge \text{List } \text{next } q \ Qs \wedge (\text{set } Ps \cap \text{set } Qs) = \{\} \wedge \text{size } Qs \leq \text{size } Ps \}$

$pp := p;$

$\text{INV } \{ \exists PPPs \ QQs \ PPPs.$

$\text{List } \text{next } pp \ PPPs \wedge \text{List } \text{next } q \ QQs \wedge \text{Path } \text{next } p \ PPPs \ pp$

$\}$

WHILE $q \neq \text{Null}$

DO

$qq := q \rightarrow \text{next}; q \rightarrow \text{next} := pp \rightarrow \text{next}; pp \rightarrow \text{next} = q; pp := q \rightarrow \text{next}; q := qq;$

OD

$\{ \text{List } \text{next } p \ (\text{splice } Ps \ Qs) \}$

Example 8



List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*

$\{ \textit{List} \textit{nxt} \textit{p} \textit{Ps} \wedge \textit{List} \textit{nxt} \textit{q} \textit{Qs} \wedge (\textit{set} \textit{Ps} \cap \textit{set} \textit{Qs}) = \{\} \wedge \textit{size} \textit{Qs} \leq \textit{size} \textit{Ps} \}$

pp := *p*;

INV $\{ \exists \textit{PPs} \textit{QQs} \textit{PPPs}.$

$\textit{List} \textit{nxt} \textit{pp} \textit{PPs} \wedge \textit{List} \textit{nxt} \textit{q} \textit{QQs} \wedge \textit{Path} \textit{nxt} \textit{p} \textit{PPPs} \textit{pp}$
 $\wedge \textit{PPPs}@\textit{splice} \textit{PPs} \textit{QQs} = \textit{splice} \textit{Ps} \textit{Qs}$

}

WHILE *q* ≠ *Null*

DO

qq := *q* → *nxt*; *q* → *nxt* := *pp* → *nxt*; *pp* → *nxt* = *q*; *pp* := *q* → *nxt*; *q* := *qq*;

OD

$\{ \textit{List} \textit{nxt} \textit{p} (\textit{splice} \textit{Ps} \textit{Qs}) \}$

Example 8



List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*

$\{ \textit{List} \textit{nxt} \textit{p} \textit{Ps} \wedge \textit{List} \textit{nxt} \textit{q} \textit{Qs} \wedge (\textit{set} \textit{Ps} \cap \textit{set} \textit{Qs}) = \{\} \wedge \textit{size} \textit{Qs} \leq \textit{size} \textit{Ps} \}$

$pp := p;$

$\text{INV} \{ \exists PPPs \textit{QQs} \textit{PPPs}. \textit{size} \textit{QQs} \leq \textit{size} \textit{PPs} \wedge$

$\textit{List} \textit{nxt} \textit{pp} \textit{PPs} \wedge \textit{List} \textit{nxt} \textit{q} \textit{QQs} \wedge \textit{Path} \textit{nxt} \textit{p} \textit{PPPs} \textit{pp}$

$\wedge \textit{PPPs}@\textit{splice} \textit{PPs} \textit{QQs} = \textit{splice} \textit{Ps} \textit{Qs}$

$\}$

WHILE $q \neq \textit{Null}$

DO

$qq := q \rightarrow \textit{nxt}; q \rightarrow \textit{nxt} := pp \rightarrow \textit{nxt}; pp \rightarrow \textit{nxt} = q; pp := q \rightarrow \textit{nxt}; q := qq;$

OD

$\{ \textit{List} \textit{nxt} \textit{p} (\textit{splice} \textit{Ps} \textit{Qs}) \}$

Example 8



List *nxt* *p* *Ps* = *Path* *nxt* *p* *Ps* *Null*

Path *nxt* *p* *Ps* *Null* is a linked list from *p* to *q* following function *nxt* and containing list of pointers *Ps*

$\{ \textit{List } \textit{nxt } p \textit{ Ps} \wedge \textit{List } \textit{nxt } q \textit{ Qs} \wedge (\textit{set } \textit{Ps} \cap \textit{set } \textit{Qs}) = \{\} \wedge \textit{size } \textit{Qs} \leq \textit{size } \textit{Ps} \}$

$pp := p;$

$\text{INV } \{ \exists PPPs \textit{ QQs} \textit{ PPPs}. \textit{size } \textit{QQs} \leq \textit{size } \textit{PPs} \wedge$

$\textit{List } \textit{nxt } pp \textit{ PPs} \wedge \textit{List } \textit{nxt } q \textit{ QQs} \wedge \textit{Path } \textit{nxt } p \textit{ PPPs } pp$

$\wedge \textit{PPPs}@\textit{splice } \textit{PPs } \textit{QQs} = \textit{splice } \textit{Ps } \textit{Qs} \wedge$

$\textit{set } \textit{PPs} \cap \textit{set } \textit{QQs} = \{\} \wedge \textit{distinct } \textit{PPPs} \wedge \textit{set } \textit{PPPs} \cap (\textit{set } \textit{PPs} \cup \textit{set } \textit{QQs}) = \{\}$

$\}$

WHILE $q \neq \textit{Null}$

DO

$qq := q \rightarrow \textit{nxt}; q \rightarrow \textit{nxt} := pp \rightarrow \textit{nxt}; pp \rightarrow \textit{nxt} = q; pp := q \rightarrow \textit{nxt}; q := qq;$

OD

$\{ \textit{List } \textit{nxt } p (\textit{splice } \textit{Ps } \textit{Qs}) \}$

A background pattern of white hexagons on a teal background, arranged in a staggered grid.

DATA
61



Demo