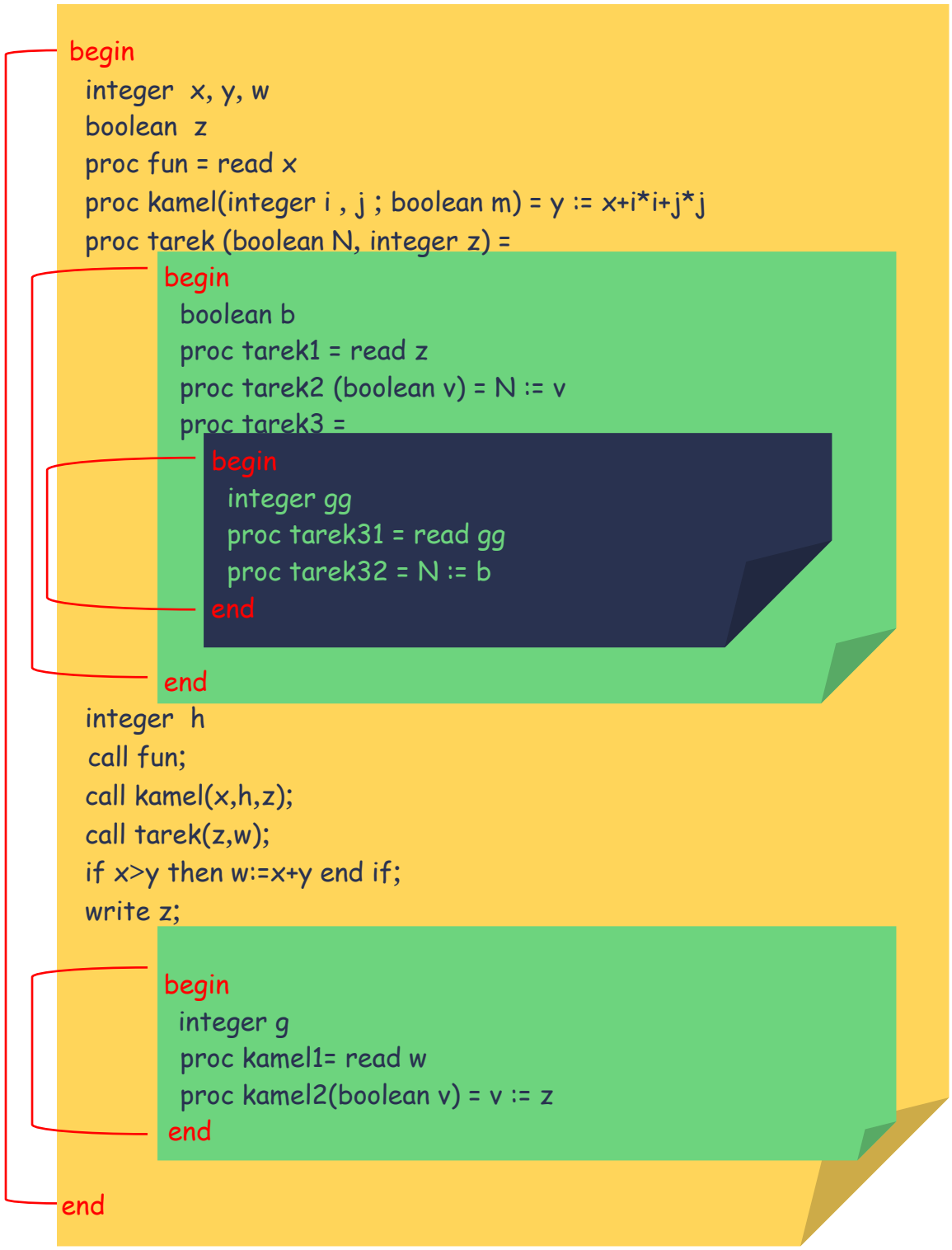
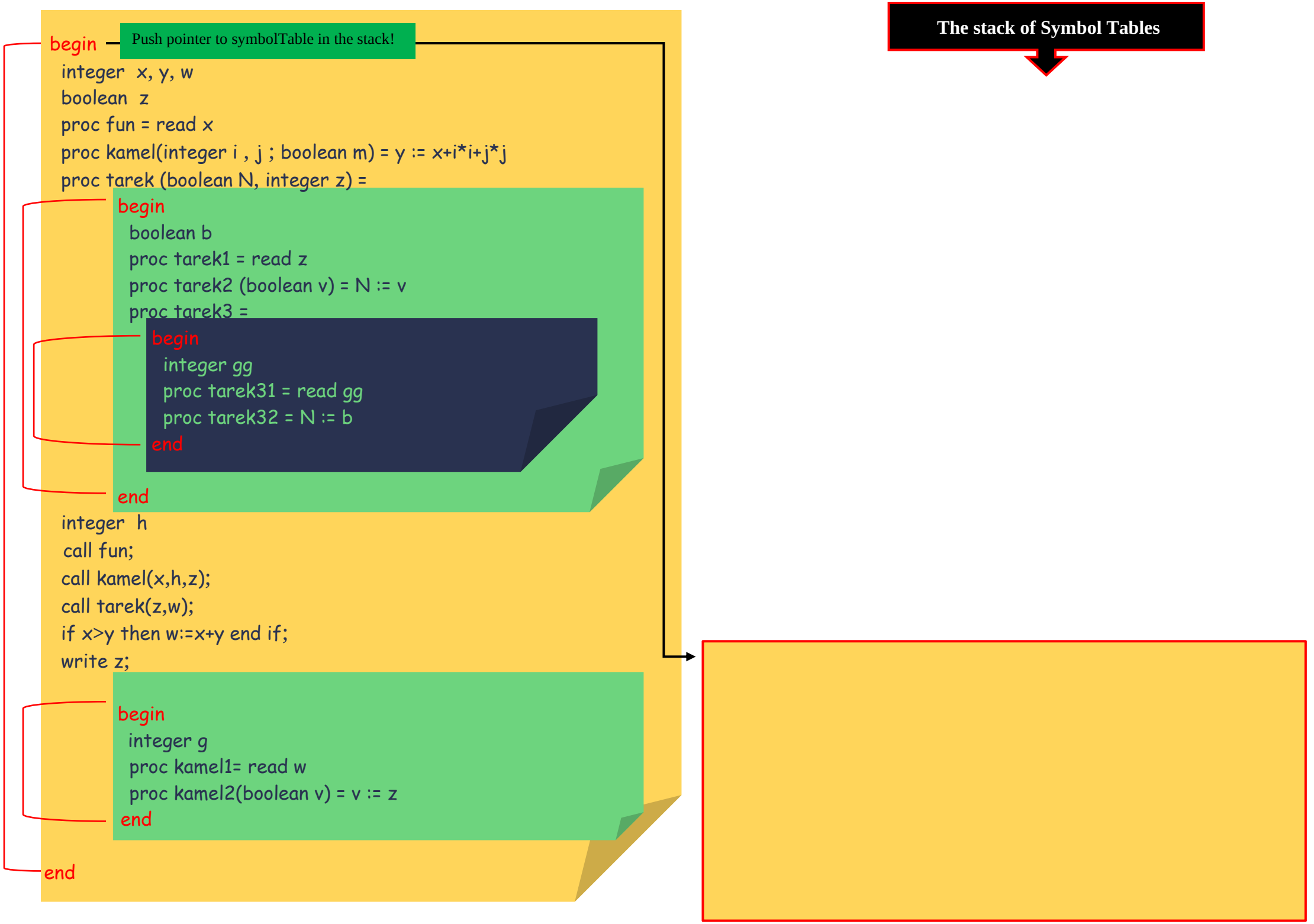


The stack of Symbol Tables


Push pointer to symbolTable in the stack!
The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end

```
begin
  integer  x, y, w
  boolean  z
  proc fun = read x
  proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
  proc tarek (boolean N, integer z) =
        begin
          boolean b
          proc tarek1 = read z
          proc tarek2 (boolean v) = N := v
          proc tarek3 =
              begin
                integer gg
                proc tarek31 = read gg
                proc tarek32 = N := b
              end
        end
  integer  h
  call fun;
  call kamel(x,h,z);
  call tarek(z,w);
  if x>y then w:=x+y end if;
  write z;
        begin
          integer g
          proc kamel1= read w
          proc kamel2(boolean v) = v := z
        end
end
```

The stack of Symbol Tables

1) <y|integer|variable>

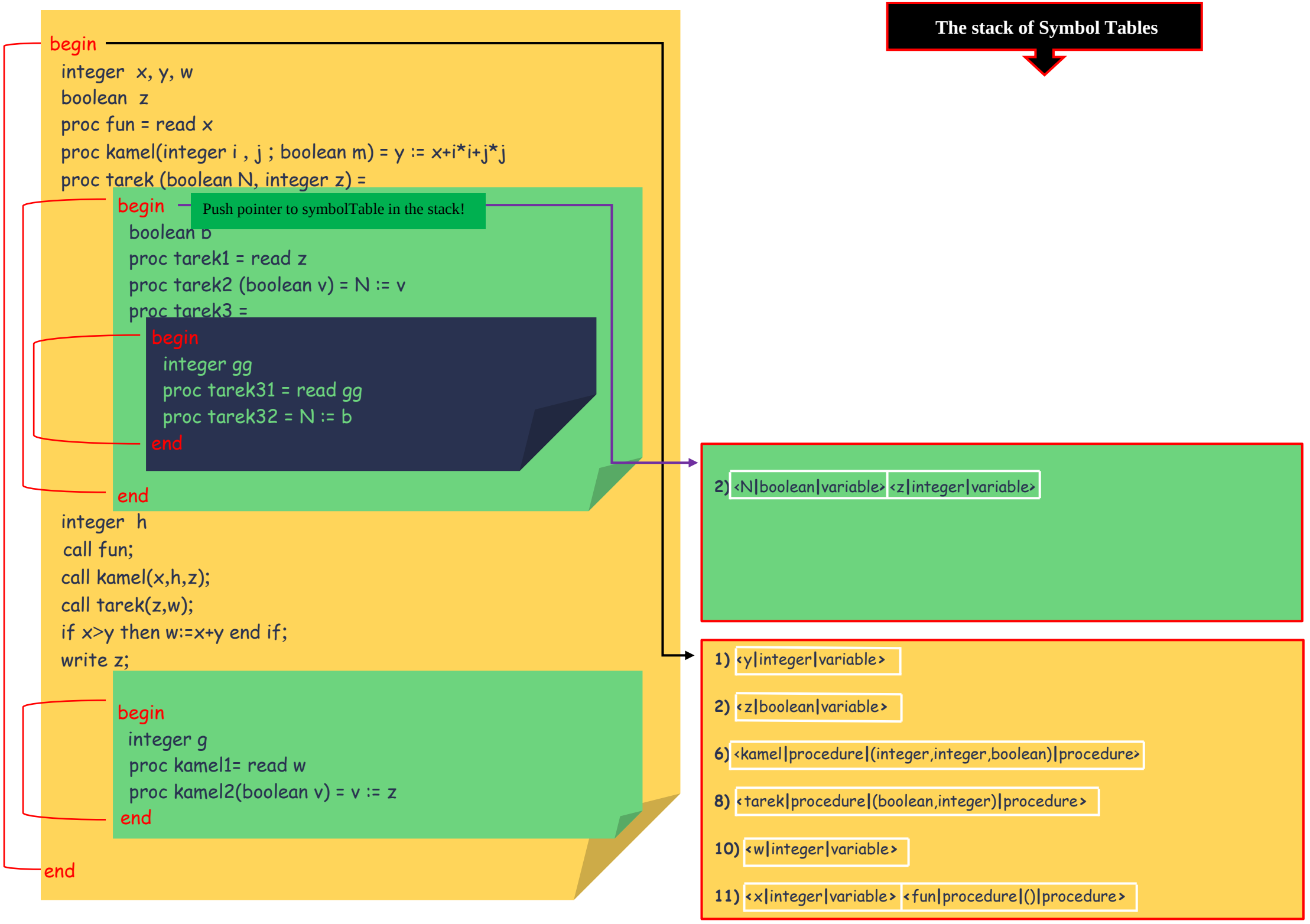
2) <z|boolean|variable>

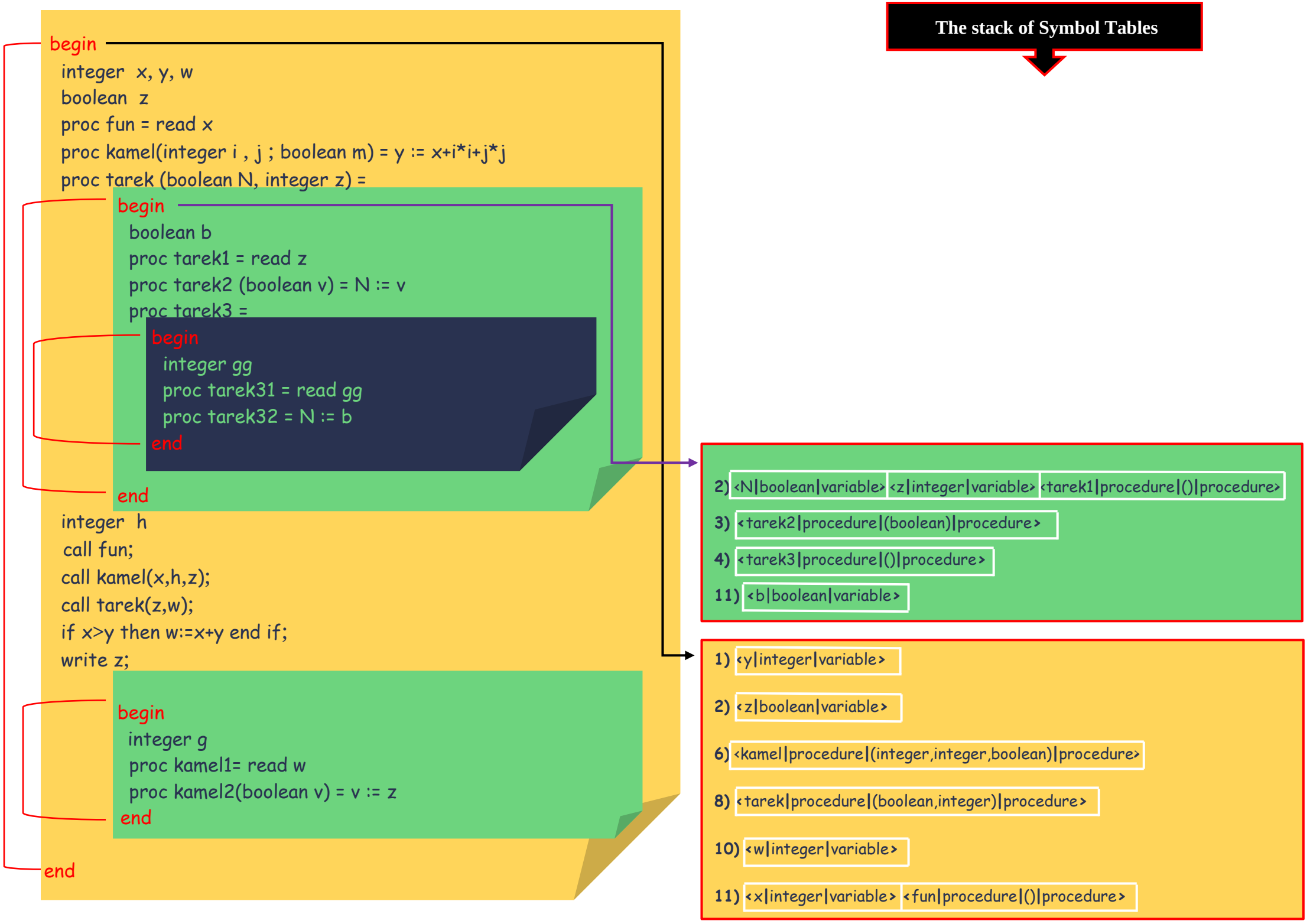
6) <kamel|procedure|(integer,integer,boolean)|procedure>

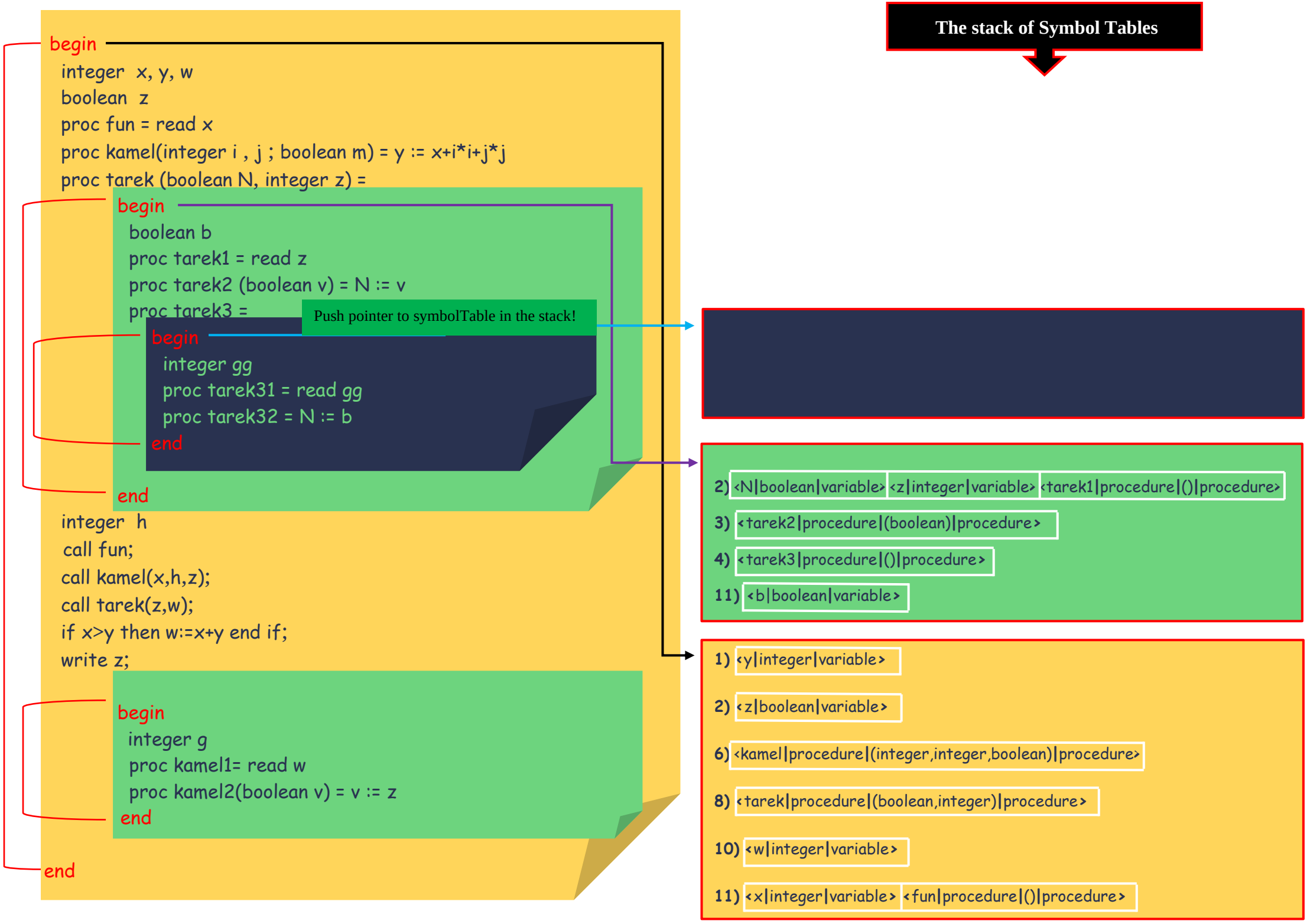
8) <tarek|procedure|(boolean,integer)|procedure>

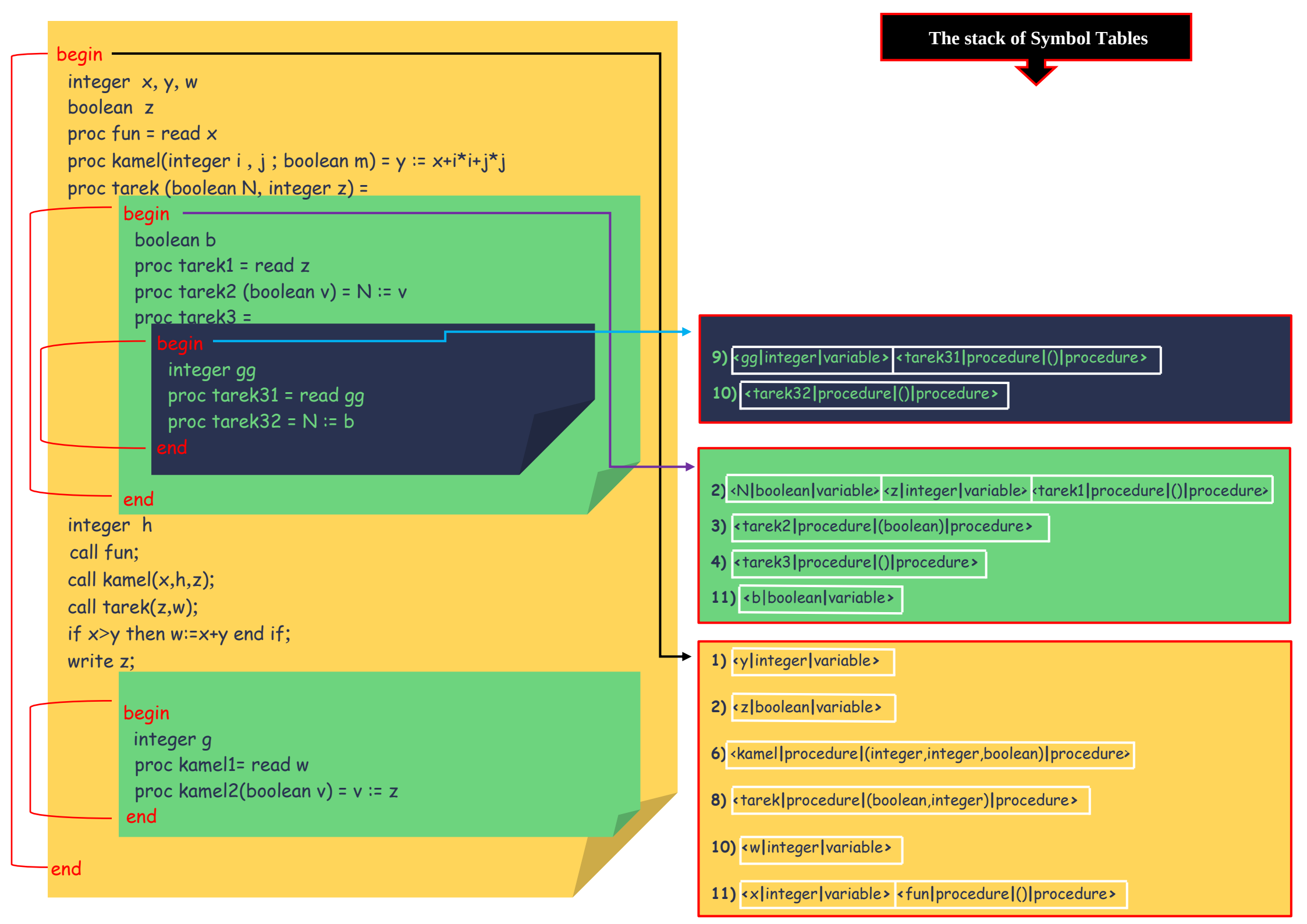
10) <w|integer|variable>

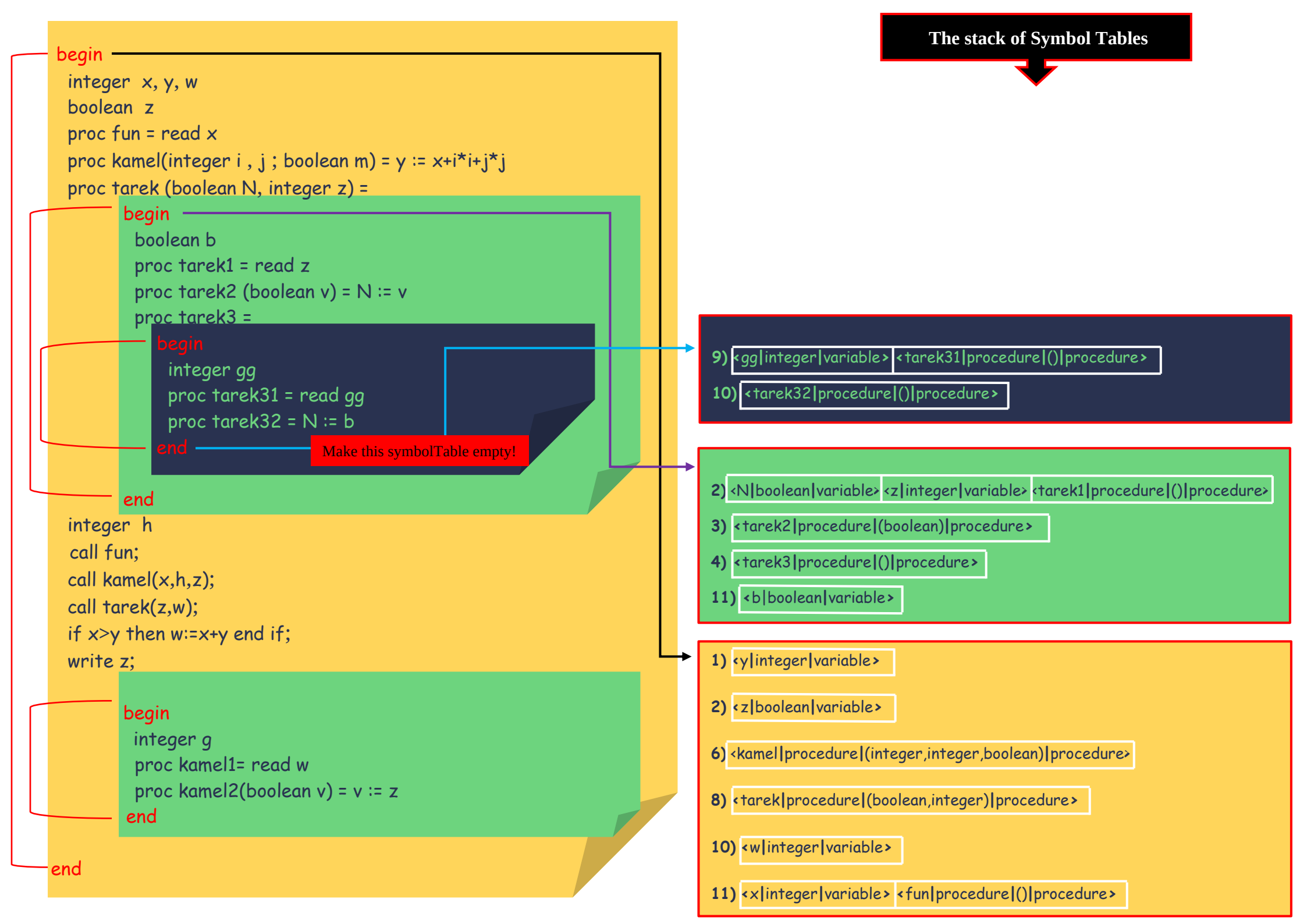
11) <x|integer|variable> <fun|procedure|()|procedure>

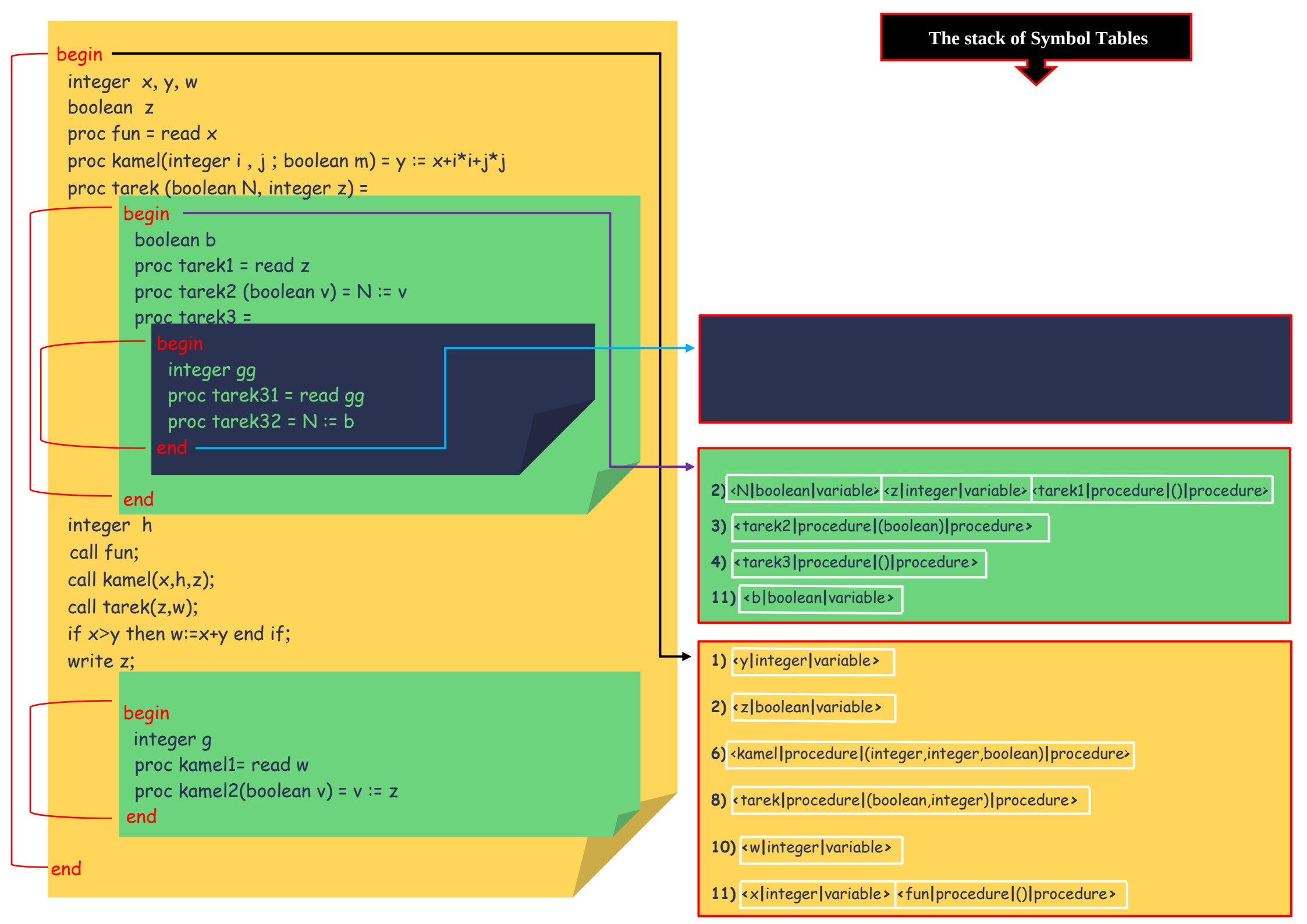

The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
Push pointer to symbolTable in the stack!
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
2) <N|boolean|variable> <z|integer|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>

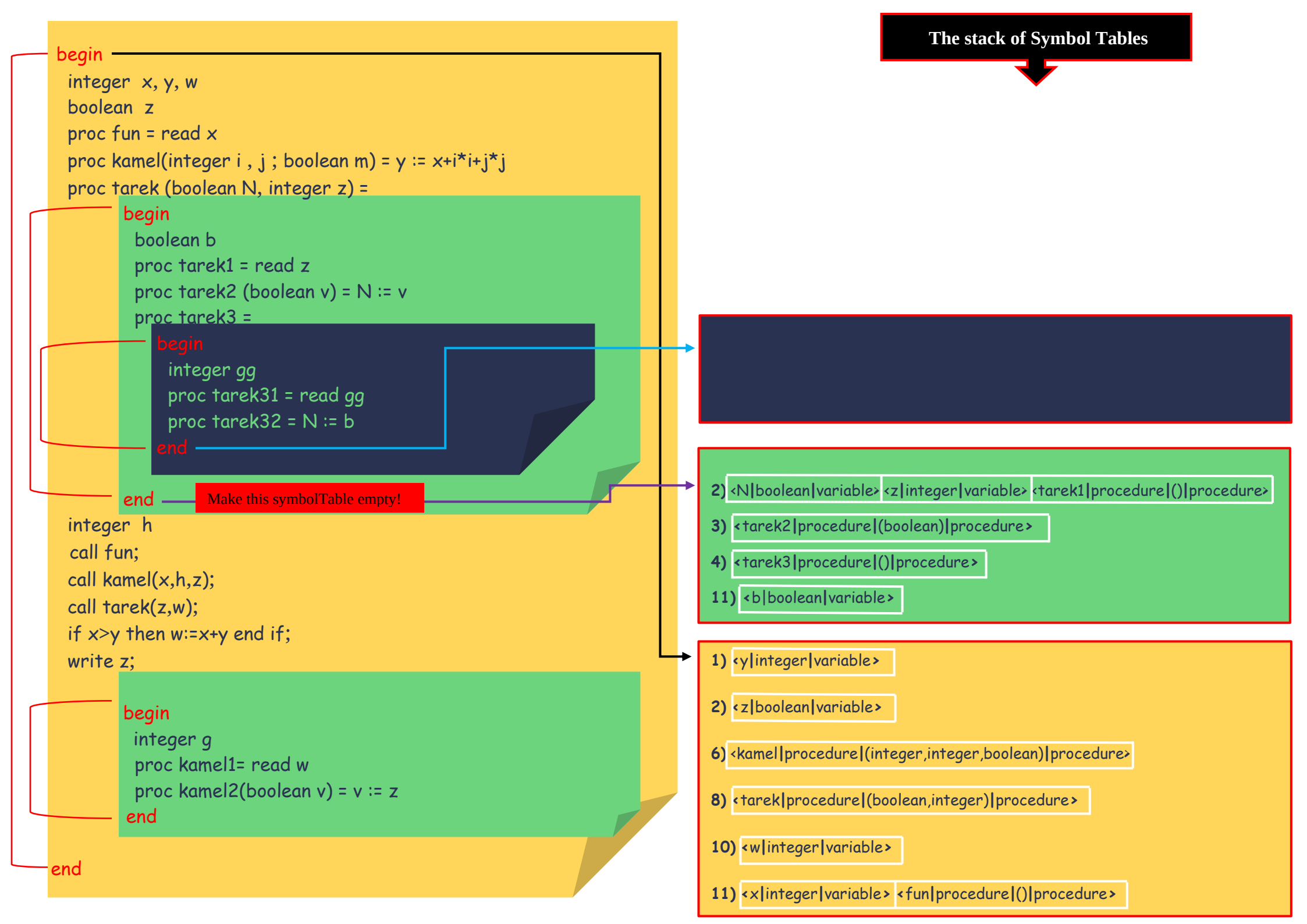

The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>

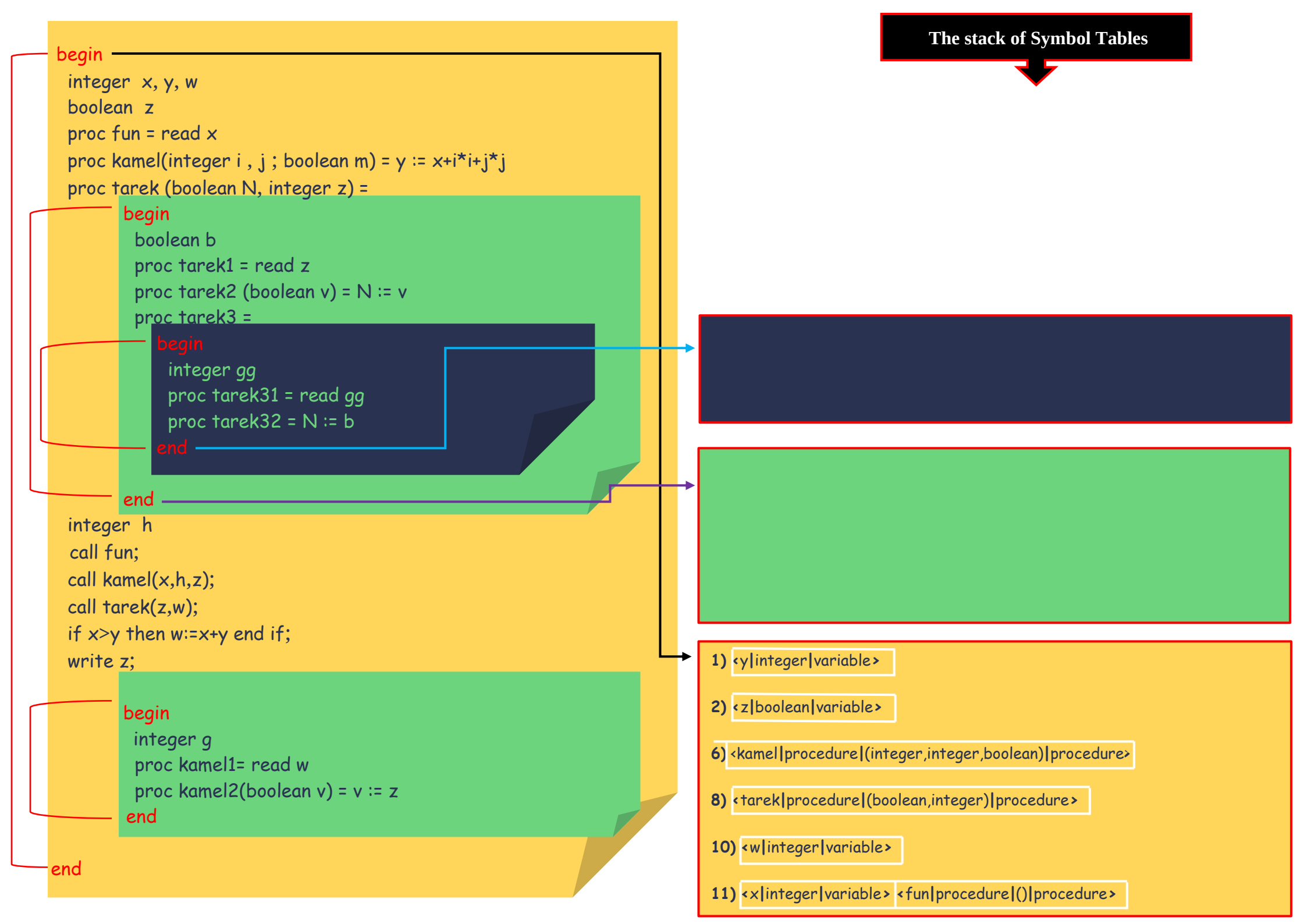

The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
Push pointer to symbolTable in the stack!
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
9) <gg|integer|variable> <tarek31|procedure|()|procedure>
10) <tarek32|procedure|()|procedure>
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
Make this symbolTable empty!
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
9) <gg|integer|variable> <tarek31|procedure|()|procedure>
10) <tarek32|procedure|()|procedure>
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
Make this symbolTable empty!
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
2) <N|boolean|variable> <z|integer|variable> <tarek1|procedure|()|procedure>
3) <tarek2|procedure|(boolean)|procedure>
4) <tarek3|procedure|()|procedure>
11) <b|boolean|variable>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>

```
begin
  integer  x, y, w
  boolean  z
  proc fun = read x
  proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
  proc tarek (boolean N, integer z) =
    begin
      boolean b
      proc tarek1 = read z
      proc tarek2 (boolean v) = N := v
      proc tarek3 =
        begin
          integer gg
          proc tarek31 = read gg
          proc tarek32 = N := b
        end
    end
  integer  h
  call fun;
  call kamel(x,h,z);
  call tarek(z,w);
  if x>y then w:=x+y end if;
  write z;
    begin
      integer g
      proc kamel1= read w
      proc kamel2(boolean v) = v := z
    end
end
```

The stack of Symbol Tables

1) <y|integer|variable>

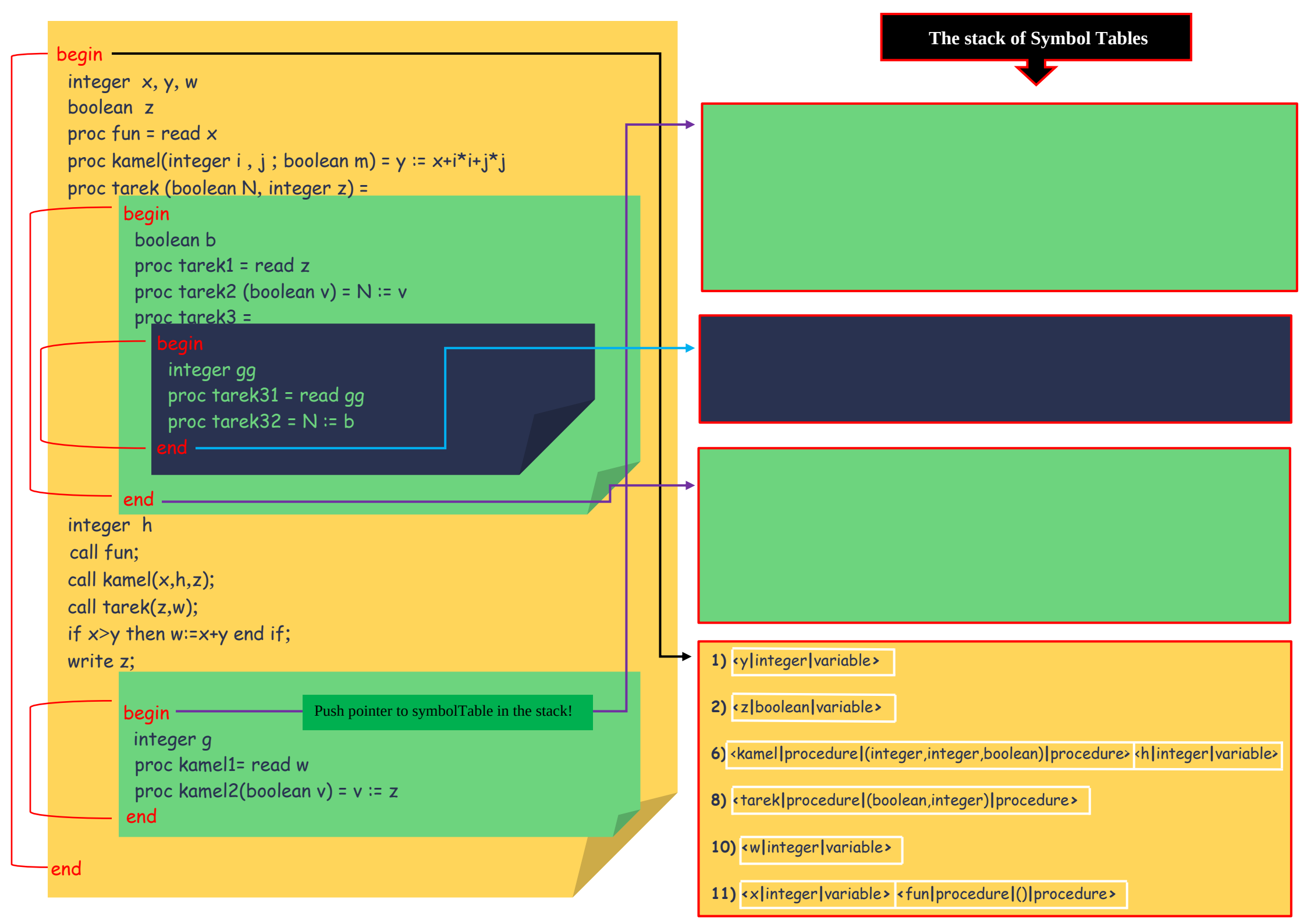
2) <z|boolean|variable>

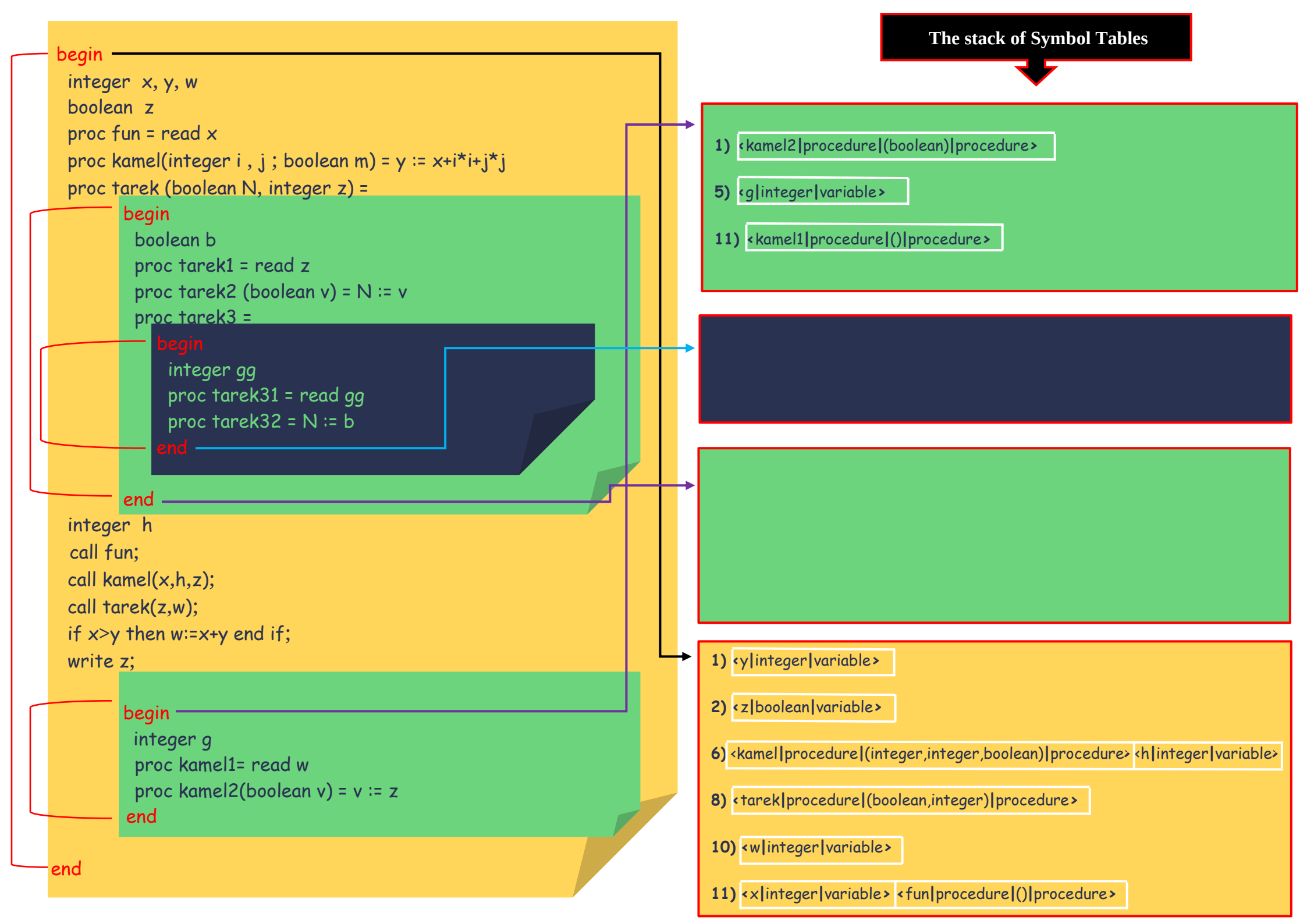
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>

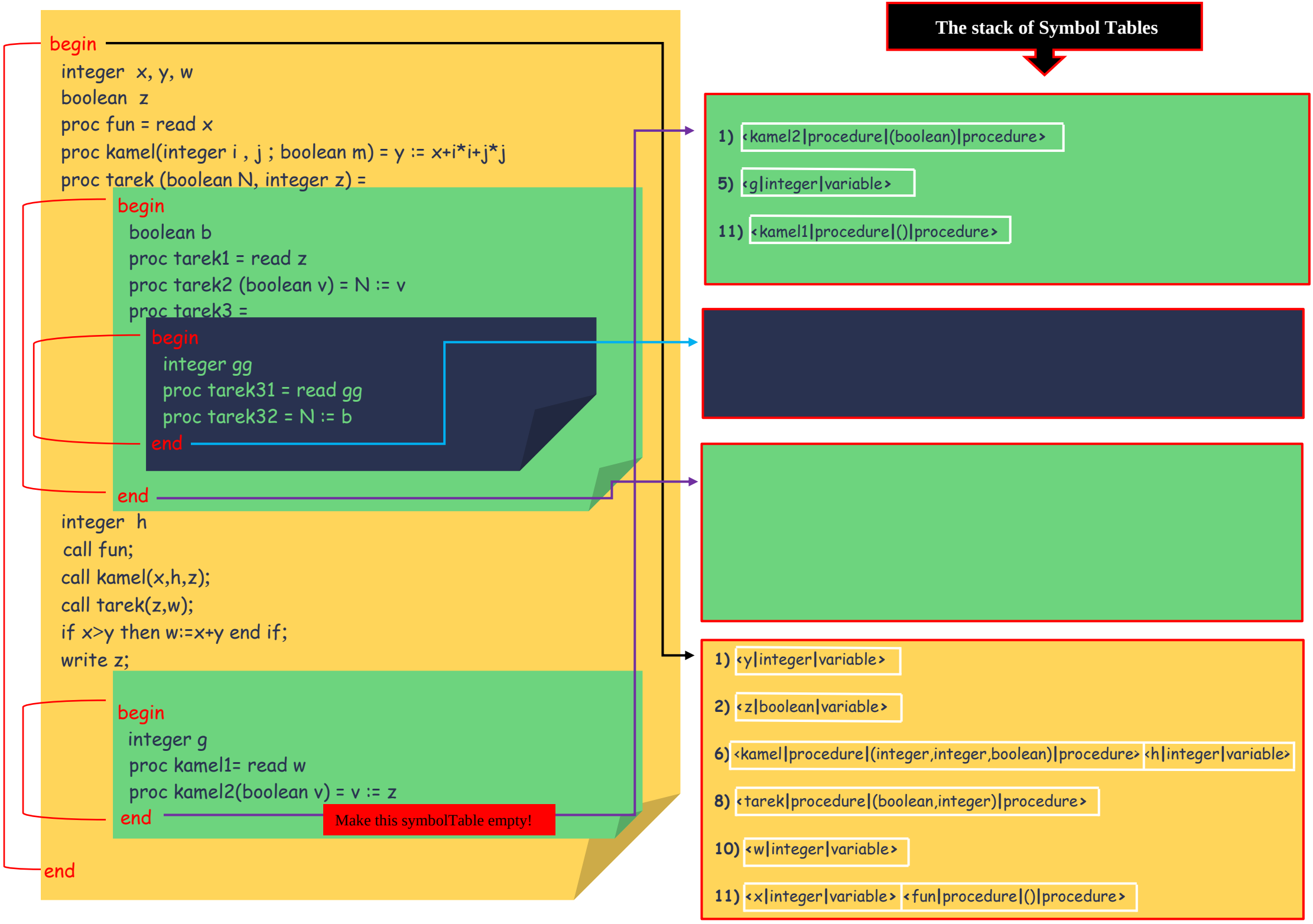
8) <tarek|procedure|(boolean,integer)|procedure>

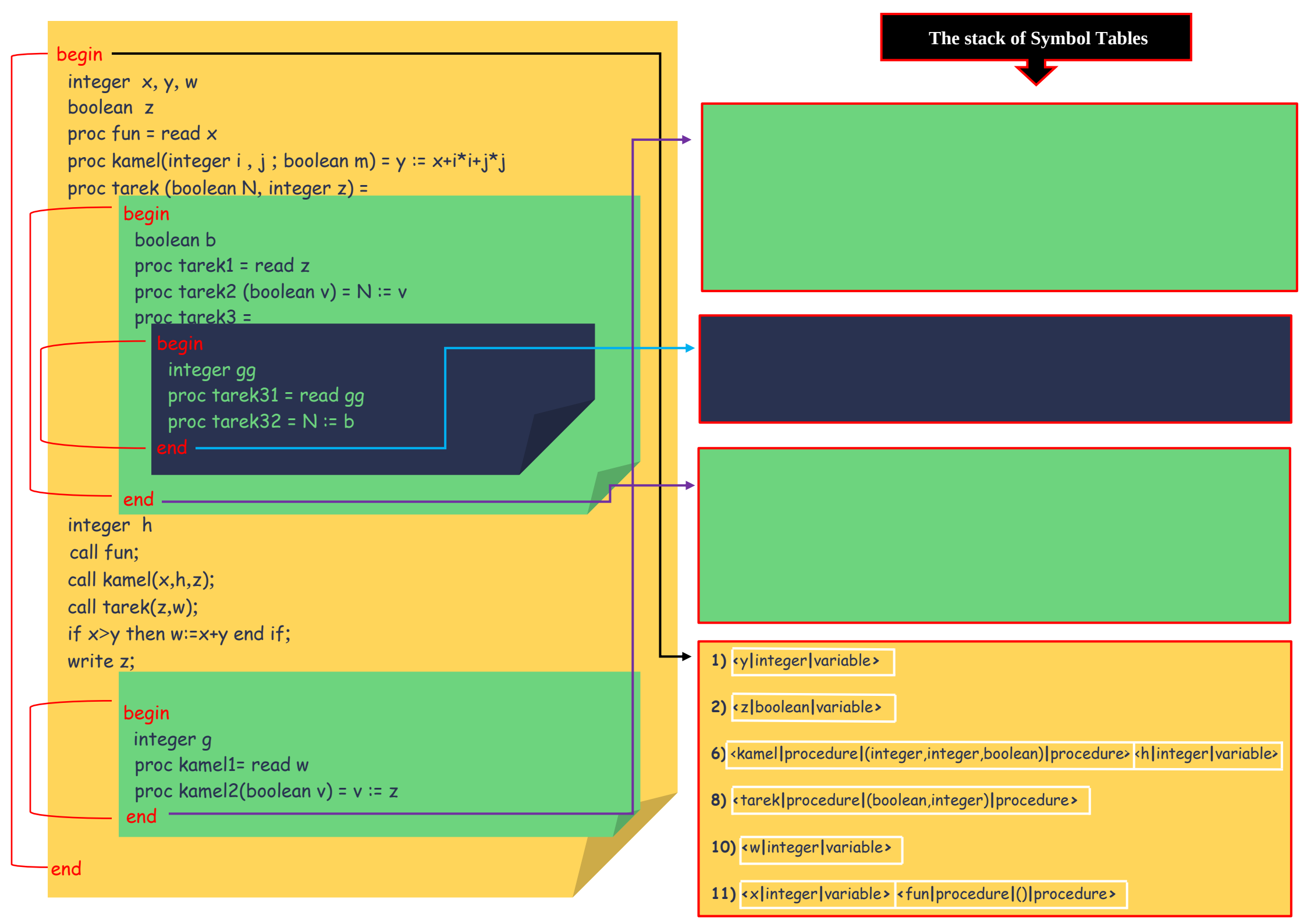
10) <w|integer|variable>

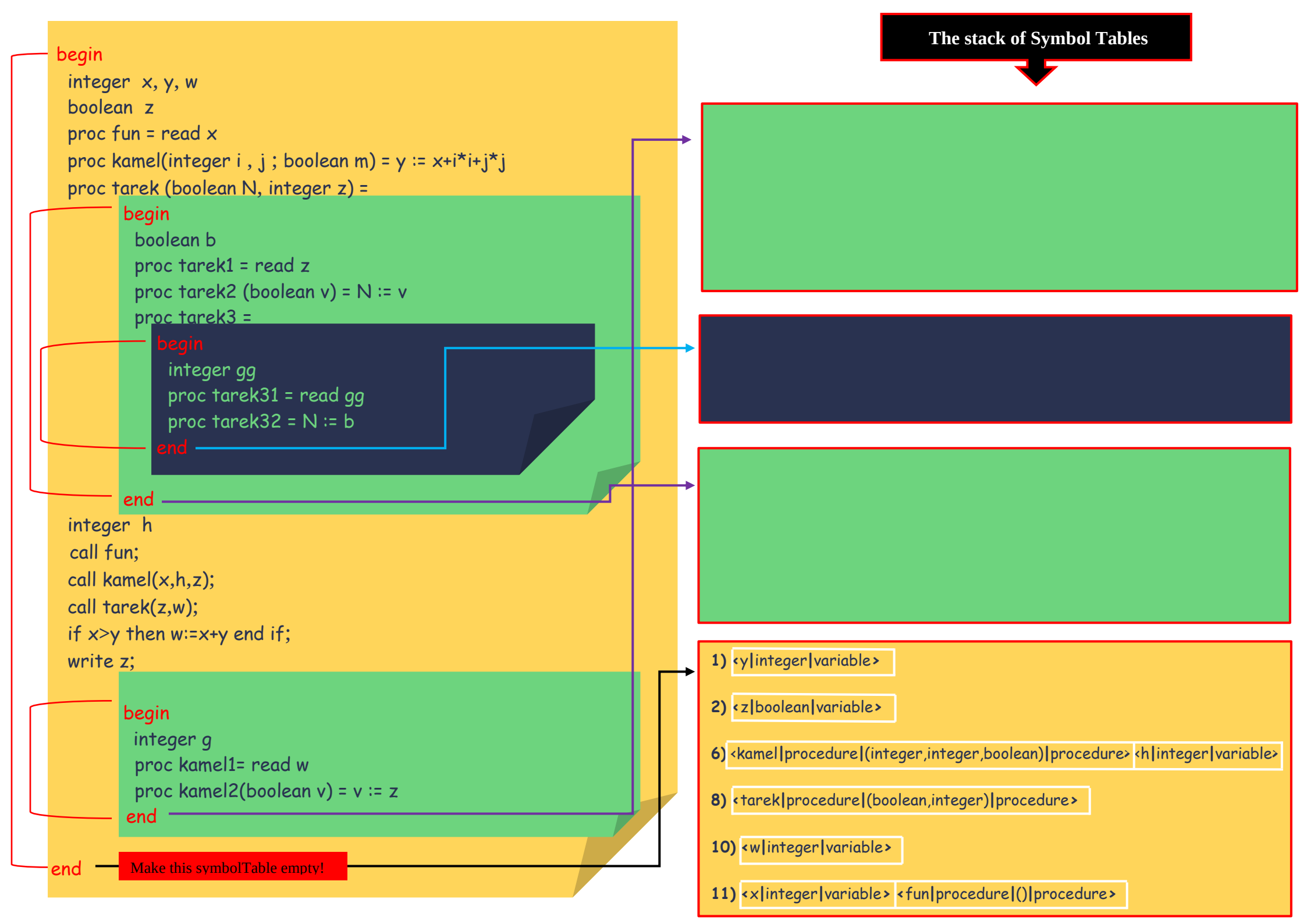
11) <x|integer|variable> <fun|procedure|()|procedure>

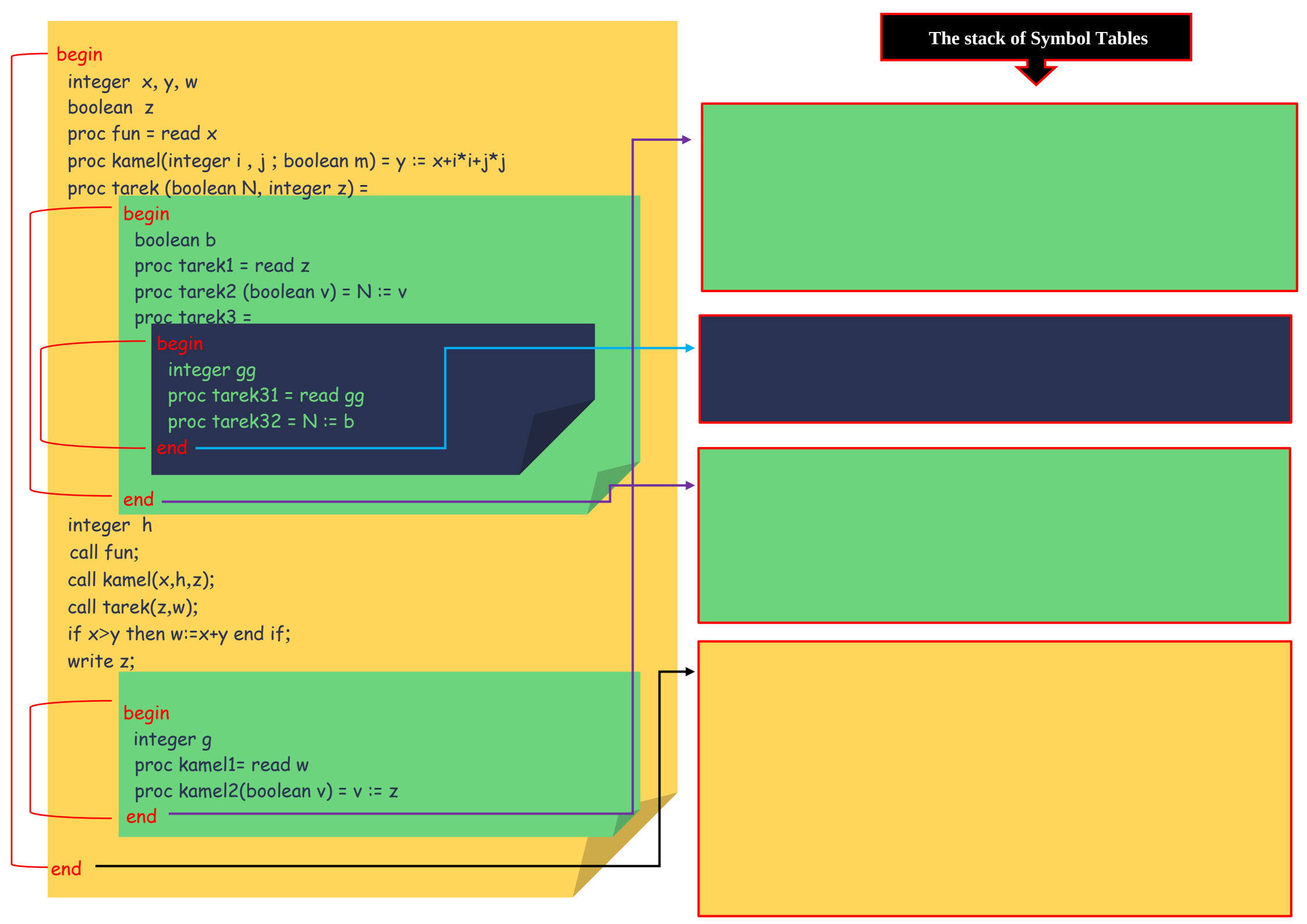

The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
Push pointer to symbolTable in the stack!
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
The stack of Symbol Tables
1) <kamel2|procedure|(boolean)|procedure>
5) <g|integer|variable>
11) <kamel1|procedure|()|procedure>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
Make this symbolTable empty!
end
The stack of Symbol Tables
1) <kamel2|procedure|(boolean)|procedure>
5) <g|integer|variable>
11) <kamel1|procedure|()|procedure>
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end
Make this symbolTable empty!
1) <y|integer|variable>
2) <z|boolean|variable>
6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
8) <tarek|procedure|(boolean,integer)|procedure>
10) <w|integer|variable>
11) <x|integer|variable> <fun|procedure|()|procedure>


The stack of Symbol Tables
begin
integer x, y, w
boolean z
proc fun = read x
proc kamel(integer i , j ; boolean m) = y := x+i*i+j*j
proc tarek (boolean N, integer z) =
begin
boolean b
proc tarek1 = read z
proc tarek2 (boolean v) = N := v
proc tarek3 =
begin
integer gg
proc tarek31 = read gg
proc tarek32 = N := b
end
end
integer h
call fun;
call kamel(x,h,z);
call tarek(z,w);
if x>y then w:=x+y end if;
write z;
begin
integer g
proc kamel1= read w
proc kamel2(boolean v) = v := z
end
end