```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
       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 q
        proc kamel1= read w
        proc kamel2(boolean v) = v := z
        end
```

The stack of Symbol Tables

```
begin — Push pointer to symbolTable in the stack!
 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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
       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 q
         proc kamel1= read w
         proc kamel2(boolean v) = v := z
        end
```

## The stack of Symbol Tables

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
       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
```

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

10) <w|integer|variable>

```
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 p
         proc tarek1 = read z
         proc tarek2 (boolean v) = N := v
         proc tarek3 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
       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
```

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
       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
```

- 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
```

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

integer x, y, w

proc fun = read x

proc kamel(integer i, j; boolean m) = y := x+i\*i+j\*j

boolean z

```
The stack of Symbol Tables
```

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

integer x, y, w

```
The stack of Symbol Tables
```

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

```
The stack of Symbol Tables
```

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

integer x, y, w

proc fun = read x

begin

boolean b

proc kamel(integer i, j; boolean m) = y := x+i\*i+j\*j

proc tarek (boolean N, integer z) =

boolean z

```
The stack of Symbol Tables
```

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

integer x, y, w

proc fun = read x

begin

boolean b

proc kamel(integer i, j; boolean m) = y := x+i\*i+j\*j

proc tarek (boolean N, integer z) =

boolean z

```
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>

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
 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
```

```
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>
```

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
 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
```

```
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 =
             integer gg
             proc tarek31 = read gg
             proc tarek32 = N := b
 integer h
 call fun;
 call kamel(x,h,z);
 call tarek(z,w);
 if x>y then w:=x+y end if;
                                                                                1) <y|integer|variable>
 write z:
                                                                                2) <z|boolean|variable>
                              Push pointer to symbolTable in the stack!
        begin
         integer q
                                                                                6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
         proc kamel1= read w
         proc kamel2(boolean v) = v := z
                                                                                8) <tarek|procedure|(boolean,integer)|procedure>
        end
                                                                                10) <w|integer|variable>
                                                                                11) <x|integer|variable> <fun|procedure|()|procedure>
```

```
The stack of Symbol Tables
1) <a href="mailto:kamel2">(kamel2">procedure</a>)
5) <glinteger|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>

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
 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
```

```
The stack of Symbol Tables
1) <a href="mailto:kamel2">kamel2</a> | procedure >
5) <glinteger|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>
```

```
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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
 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!
```

```
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 =
             integer gg
             proc tarek31 = read gg
             proc tarek32 = N := b
 integer h
 call fun;
 call kamel(x,h,z);
 call tarek(z,w);
 if x>y then w:=x+y end if;
                                                                               1) <y|integer|variable>
 write z:
                                                                               2) <z|boolean|variable>
       begin
         integer g
                                                                               6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
         proc kamel1= read w
         proc kamel2(boolean v) = v := z
                                                                               8) <tarek|procedure|(boolean,integer)|procedure>
        end -
                                                                               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 =
             integer gg
             proc tarek31 = read gg
             proc tarek32 = N := b
 integer h
 call fun;
 call kamel(x,h,z);
 call tarek(z,w);
 if x>y then w:=x+y end if;
                                                                                1) <y|integer|variable>
 write z:
                                                                                2) <z|boolean|variable>
       begin
         integer g
                                                                                6) <kamel|procedure|(integer,integer,boolean)|procedure> <h|integer|variable>
         proc kamel1= read w
         proc kamel2(boolean v) = v := z
                                                                                8) <tarek|procedure|(boolean,integer)|procedure>
        end -
                                                                                10) <w|integer|variable>
        Make this symbolTable empty!
                                                                                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 =
            integer gg
            proc tarek31 = read gg
            proc tarek32 = N := b
 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 q
        proc kamel1= read w
        proc kamel2(boolean v) = v := z
        end -
```