The example instruction in kernel language

```
local X in
   local B in
      B=true
      if B then X=1 else skip end
   end
end
```

Start of the execution: the initial execution state

```
([(local X in
    local B in
      B=true
      if B then X=1 else skip end
    end
  end, {})],
 {})
```

- We start with an empty memory and an empty environment

The *local X in ... end* instruction

```
([(local B in
      B=true
      if B then X=1 else skip end
   end,
   {X → x})],
 {x})
```

- We create a new variable *x* in memory
- We put the inner instruction on the stack and add X→*x* to its environment

The *local B in ... end* instruction

```
([( (B=true
      if B then X=1 else skip end) ,
    {B → b, X → x})] ,
 {b,x})
```

- We create a new variable *b* in memory
- We put the inner instruction on the stack and add B→*b* to its environment

The sequential composition instruction

```
([(B=true,{B → b, X → x}),
   (if B then X=1
    else skip end,{B → b, X → x})],
 {b,x})
```

- We split the sequential composition into its two parts
- We put the two instructions on the stack
- The environments stay the same

The *B=true* instruction

([(**if** B **then** X=1
    else skip end,{B → *b*, X → *x*})],
 {*b*=**true**, *x*})

- We bind *b* to **true** in memory

The conditional instruction

([(X=1,{B → *b*, X → *x*})],
{*b*=**true**, *x*})

- We read the value of B
- Since B is **true**, it puts the instruction after **then** on the stack
- If B is **false**, it will put the instruction after **else** on the stack
- If B has any other value, then the conditional raises an error

- (Note: If B is unbound then the execution of the semantic stack stops until B becomes bound – this can only happen in another semantic stack, i.e., with concurrency)

The *X=1* instruction

([],
{*b*=**true**, *x*=1})

- We bind *x* to 1 in memory
- Execution stops because the stack is empty

Semantic rules we have seen

- This example has shown us the execution of four instructions:
  - **local** <x> **in** <s> **end** (variable creation)
  - <s>$_1$ <s>$_2$ (sequential composition)
  - **if** <*x*> **then** <s>$_1$ **else** <s>$_2$ **end** (conditional)
  - <x>=<v> (assignment)
- In the next unit we will see the semantic rules corresponding to these instructions