Procedure value

- Procedure declarations look like statements:

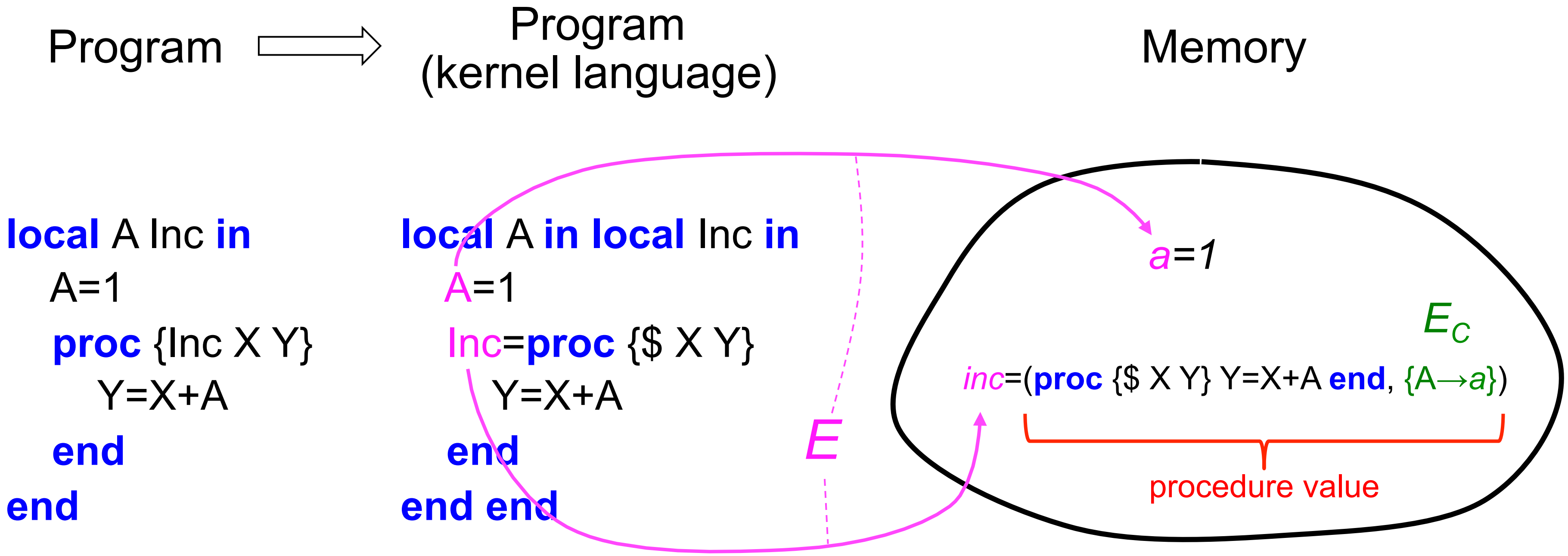
```
proc {Inc X Y} Y=X+A end
```

- But this is syntactic sugar! What really happens is that the identifier Inc refers to a variable that is bound to a procedure value:

```
Inc=proc {$ X Y} Y=X+A end
```

- The $ symbol is a placeholder to show that the procedure definition has no identifier. Instead of just removing the identifier, we replace it by a new symbol that cannot be confused with an identifier.

How procedures are stored in memory



Program ⇒ Program (kernel language)

Memory

```
local A Inc in
   A=1
   proc {Inc X Y}
      Y=X+A
   end
end
```

```
local A in local Inc in
   A=1
   Inc=proc {$ X Y}
      Y=X+A
   end
end end
```

a=*1*

E_C

inc=(**proc** {$ X Y} Y=X+A **end**, {A→*a*})

procedure value

E = {A→*a*,Inc→*inc*}

Procedure values

- A procedure value is stored in memory as a pair:

$$\mathit{inc} = (\underbrace{\textbf{proc}\ \{\$\ X\ Y\}\ Y{=}X{+}A\ \textbf{end}}_{\text{Procedure code}},\ \underbrace{\{A \longrightarrow \mathit{a}\}}_{\text{Contextual environment}})$$

- The variable *inc* is bound to the procedure value
  - Terminology: a procedure value is also called a *closure* or a *lexically scoped closure*