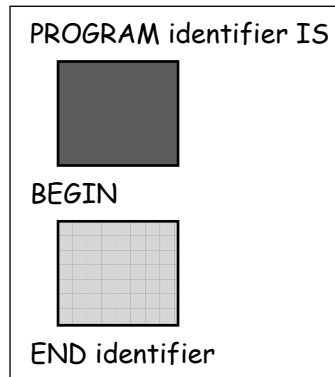


## *BL Program*

---



## *Math Model for Program*

---

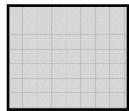
- What pieces of information do we need to keep track of for a BL program?

## *The Math Model*

- **math subtype PROGRAM is (**  
    name: IDENTIFIER  
    context: CONTEXT  
    body: STATEMENT  
  )  
  **exemplar p**  
  **constraint**  
    root (p.body).kind = BLOCK

## *New Instruction*

INSTRUCTION identifier IS



END identifier

## *Math Model Continued...*

- What pieces of information do we need to keep track of for a new BL instruction?

## *Math Model Continued...*

- **math subtype CONTEXT is**  
**finite set of (**  
    name: IDENTIFIER  
    body: STATEMENT  
**)**  
**exemplar c**  
**constraint**  
    ...
  1. c is a (partial) function
  2. for each (name, body) pair in c,  
name is not one of primitives  
nor the empty string, and body  
is a BLOCK

## *Program Component*

- Type
  - Program\_Kernel is modeled by PROGRAM
- Initial Value
  - IS\_INITIAL\_PROGRAM (self) ≡  
self.name = empty\_string and  
self.context = empty\_set and  
IS\_INITIAL\_STATEMENT (self.body)

## *Program SteerClear*

Draw a picture of this BL program's abstract view:

```
PROGRAM SteerClear IS
  INSTRUCTION StepAside IS
    IF random THEN
      turnright
    ELSE
      turnleft
    END IF
    move
  END StepAside
  INSTRUCTION TurnAround IS
    turnright
    turnright
  END TurnAround
BEGIN
  WHILE true DO
    IF next-is-empty THEN
      skip
    ELSE
      IF next-is-wall THEN
        TurnAround
      ELSE
        StepAside
      END IF
    END IF
  END WHILE
END SteerClear
```

## *Program SteerClear (cont'd)*

p =

## *Program Continued...*

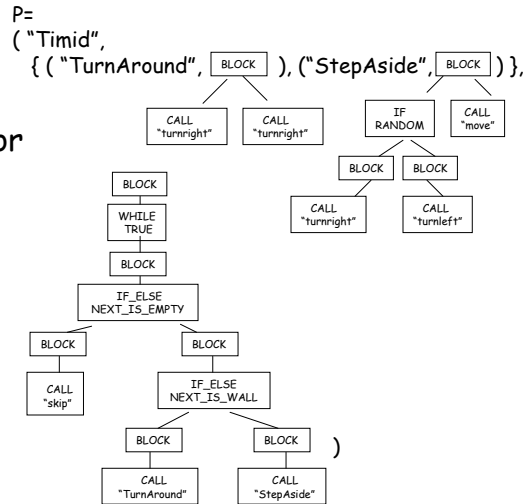
### ■ Operations

- p.Swap\_Name (name)
- p.Swap\_Body (statement)
- p.Add\_To\_Context (name, statement)
- p.Remove\_From\_Context (name, name\_copy, statement)
- p.Remove\_Any\_From\_Context (name, statement)
- p.Is\_In\_Context (name)
- p.Size\_Of\_Context ()

## Swap\_Name

What effect will the following statements have on p, the program object for program *SteerClear*?

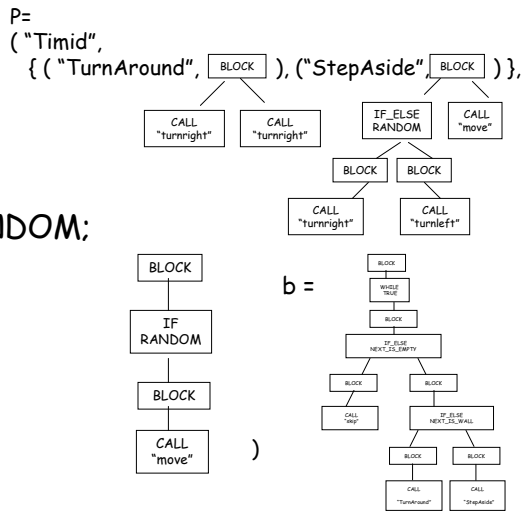
**object** Text name;  
 name = "Timid";  
 p.Swap\_Name (name);  
  
 name = "SteerClear"



## Swap\_Body

What effect will the following statements have on object p from the previous slide?

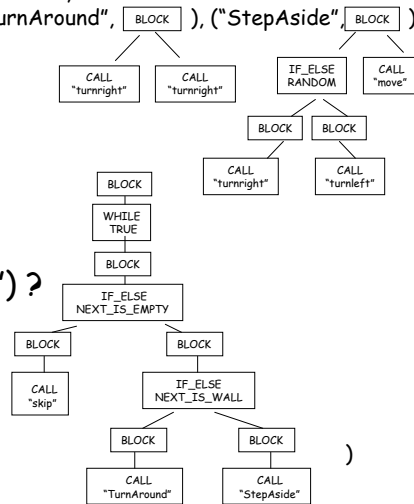
**object** Statement b, i, c;  
**object** Text m = "move";  
**object** Integer test = RANDOM;  
 c.Compose\_Call (m);  
 b.Add\_To\_Block (0, c);  
 i.Compose\_If (test, b);  
 b.Add\_to\_Block (0, i);  
 p.Swap\_Body (b);



## Context Operations

What result is produced by these statements if p has its original value?

P=  
("SteerClear",  
{ ("TurnAround", BLOCK ), ("StepAside", BLOCK ) },



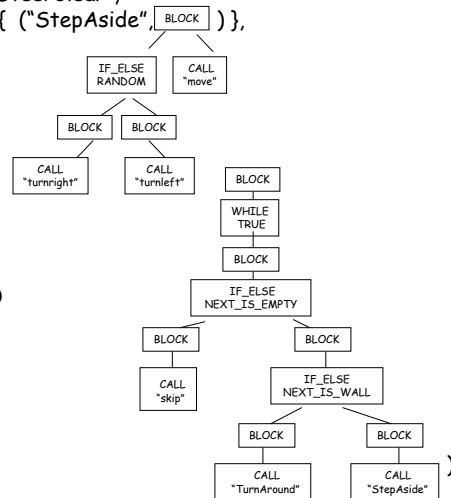
p.Size\_Of\_Context () ?

p.Is\_In\_Context ("StepAside") ?

## Context Operations (cont'd)

What result is produced by these statements if p has the value shown on the previous slide?

P=  
("SteerClear",  
{ ("StepAside", BLOCK ) },



**object** Text n;  
**object** Statement b;  
p.Remove\_From\_Context  
("TurnAround", n, b) ?

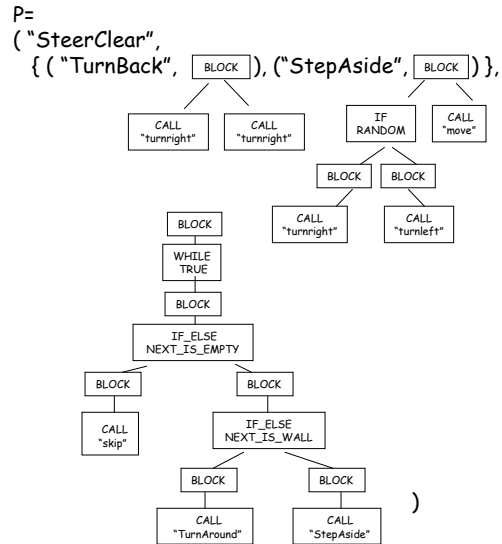
n="TurnAround"    b =

## Context Operations (cont'd)

What result is produced by these statements if p and b have the values shown on the previous slide?

n = "TurnBack";  
p.Add\_To\_Context(n, b)?

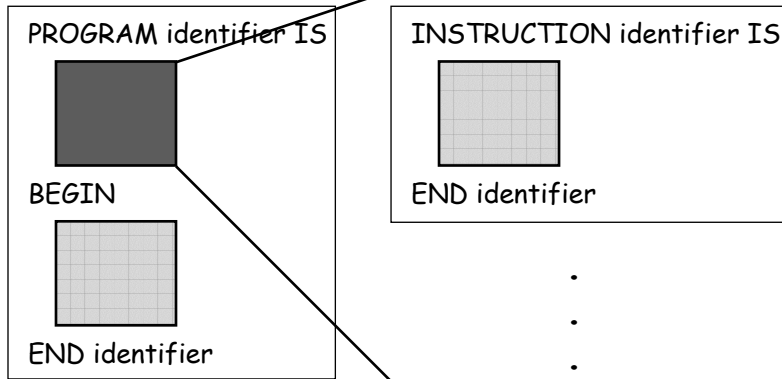
n = ""      b = BLOCK



## An Operation on Program

What would it mean to  
*demobilize* a program?

## *Statements in BL Program*



## *State the Problem*

```
global_procedure Demobilize (  
  alters Program& p  
);  
/*!  
  ensures  
    p.name = #p.name and  
    p.context =  
      CONTEXT_DEMOBILIZE (#p.context) and  
    p.body = DEMOBILIZE (#p.body)  
!*/
```

## *State the Problem Continued...*

```
math definition CONTEXT_DEMOBILIZE (  
  c: CONTEXT  
); CONTEXT satisfies  
if c = empty_set then  
  CONTEXT_DEMOBILIZE (c) = c  
else  
  there exists n: IDENTIFIER, s: STATEMENT,  
    rest_of_context: CONTEXT  
  (c = {(n, s)} union rest_of_context and  
  CONTEXT_DEMOBILIZE(c) =  
    {(n, DEMOBILIZE(s))} union  
    CONTEXT_DEMOBILIZE (rest_of_context))
```

## *Implementation*

```
procedure_body Demobilize (  
  alters Program& p  
)  
{  
  object Program tmp;  
  object Text name;  
  object Statement body;  
  
  p.Swap_Body (body);  
  Demobilize (body);  
  tmp.Swap_Body (body);  
  
  while (p.Size_Of_Context () > 0)  
  {  
    p.Remove_Any_From_Context (name, body);  
    Demobilize (body);  
    tmp.Add_To_Context (name, body);  
  }  
  
  p.Swap_Name (name);  
  tmp.Swap_Name (name);  
  p &= tmp;  
}
```