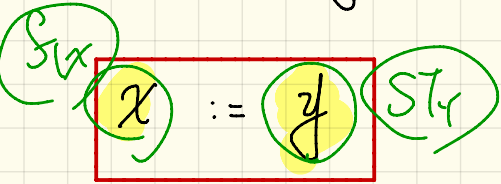


Thursday Oct. 18
Lecture 11

- Review Session

Friday 3pm LAS B

Type Checking Rules (1)



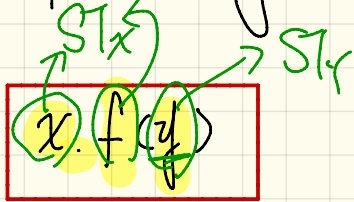
s1 : STUDENT

s2 : RS

s3 : NRS

- 1 s1 := s2 ✓
- 2 s1 := s3 ✓
- 3 s2 := s1 ✗
- 4 s3 := s1 ✗
- 5 ✓ s2 := s3 ✗
- 6 s3 := s1 ✗

Type Checking Rules (2)



```
class SMS  
  add_rs (s: RS)  
  do  
  end  
end
```

sms: SMS

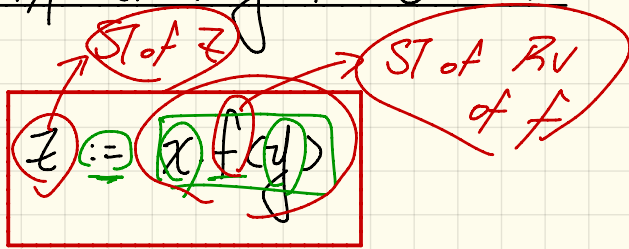
s1: STUDENT

s2: RS

s3: NRS

- STUDENT
- 1 s1. add_rs (s2) X
 - 2 X sms. add_rs (s1) \rightarrow ST: S
 - 3 \checkmark sms.add_rs (s2) \rightarrow ST: RS
 - 4 X sms.add_rs (s3) \rightarrow ST: NRS

Type Checking Rules (3)



```
class SMS
  get_S(i: INTEGER): STUDENT
  do
  ...
  end
end
```

sms: SMS

s1: STUDENT

s2: RS

s3: ~~URS~~

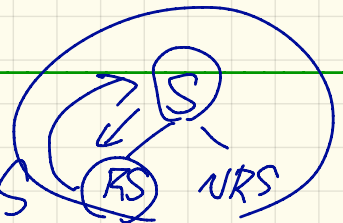
- 1 s1 := ^Ss2.get_S(1) X
- 2 ^Ss1 := sms.get_S(1) ✓
- 3 ^{RS}s2 := sms.get_S(1) X
- 4 s3 := sms.get_S(1) X
URS STUDENT

Type Checking Rules (4)

check attached {C} {Z} then
 ...
 end

ST1
 C is ancestor of ST1
 descendant of ST1

```
class SMS
  get_S(i: INTEGER) (STUDENT)
  do
  ...
  end
end
```



- sms: SMS
- sl: STUDENT
- s2: RS
- s3: NRS

- 1 check attached {RS} {S} then ... end ✓
 - 2 check attached {STUDENT} {S} then ... end ✓
 - 3 check attached {SMS} {sl} then ... end ✗
 - 4 check attached {RS} {s3} then ... end ✗
 - 5 check attached {RS} {sms get (1)} then ... end ✓
- STUDENT

Type Checking Rules (5)

```
class SMS
```

```
  get_s(i: INTEGER): STUDENT
  do
  end ...
```

```
end
```

check attached {CS} z as temp then
 → x = temp
 end STx → ST: C

sms: SMS

s1: STUDENT

s2: RS

s3: NRS

→ check attached {RS} sms.get_s(1) as temp then

1 s1 := temp RS ✓

2 s3 := temp NRS RS X

end

Type Checking Rules (6)

$\uparrow \downarrow$ $\text{ST: } C$
check attached $\{C\}$ z as temp then
 $\checkmark x.f(\text{temp})$
end

```

class SMS
  get_s(i: INTEGER): STUDENT
  do
  ...
  end
  add_rs(s: RS)
  do
  ...
  end
  
```

$[RS \text{ temp} = (RS) \text{ sms.get_s}(i);$
 $\text{sms.add_rs}(\text{temp});$

- smS: SMS
- sl: STUDENT
- s2: RS
- s3: NRS

RT. \downarrow
 S
 ST: RS
check attached $\{RS\}$ $\text{sms.get_s}(i)$ as temp then
 $\text{sms.add_rs}(\text{temp})$ ST: NRS temp.set_pr
end
 $\downarrow \text{ST: RS}$
 NRS \times
 ST: NRS
 NRS \times
check attached $\{NRS\}$ $\text{sms.get_s}(i)$ as temp then
 $\text{sms.add_rs}(\text{temp})$
end

S: STUDENT

create {RS} s. make

at RT assertion failure here

ST: NRS



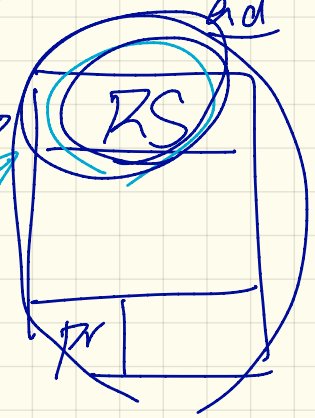
at RTs you don't reach this line

end

ST: NRS

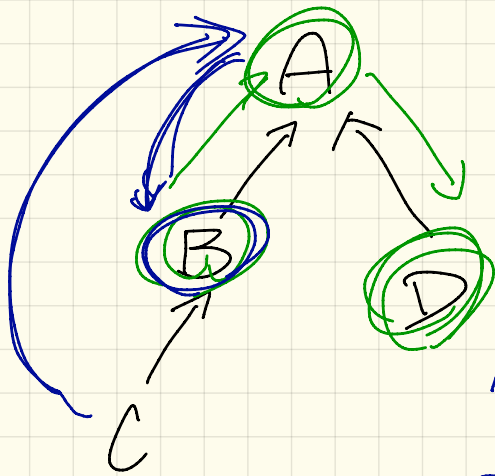
STUDENT S

NRS temp



If we allowed the cast to succeed.

- ⇒ crash when calling temp.set_drv(0.75)
- ⇒ not allowed at the cast



c: C

b: B
d: D

- 1 create {C} b. make ST: B
- 2 check attached {B} (b) as temp them
- 3 d := temp

check attached {A} c as [temp1] then B

check attached {B} temp1 as [temp2] then

end

```

    graph TD
      C((C)) --> A((A))
      A --> B((B))
      A --> T1((temp1))
      B --> T2((temp2))
  
```

check attached {A} to temp 1 and then

attached {D} temp 1 as temp 2 then

'
'
'

end

General Book

Supplier

```

class BOOK
  names: ARRAY[STRING]
  records: ARRAY[ANY]
  -- Create an empty book
  make do ... end
  -- Add a name-record pair to the book
  add(name: STRING; record: ANY) do ... end
  -- Return the record associated with a given name
  get(name: STRING): ANY do ... end
end

```

Handwritten annotations: 'DATE' circled in green above 'ANY'; 'G' written in red; 'STUDENT' written in blue; 'PART' circled in blue; 'DATE' circled in green; 'ANY' circled in green; 'DATE' circled in green; 'G' written in red.

Client

```

1 birthday: DATE; phone_number: STRING
2 b: BOOK; is_wednesday: BOOLEAN
3 create {BOOK} b.make
4 phone_number := "416-677-1010"
5 b.add("SuYeon", phone_number)
6 create {DATE} birthday.make(1975, 4, 10)
7 b.add("Yuna", birthday)
8 is_wednesday := b.get("Yuna").get_day_of_week = 4

```

Handwritten annotations: 'b: BOOK;' highlighted in cyan; 'DATE' circled in red; 'DATE' circled in blue; 'DATE' circled in blue.