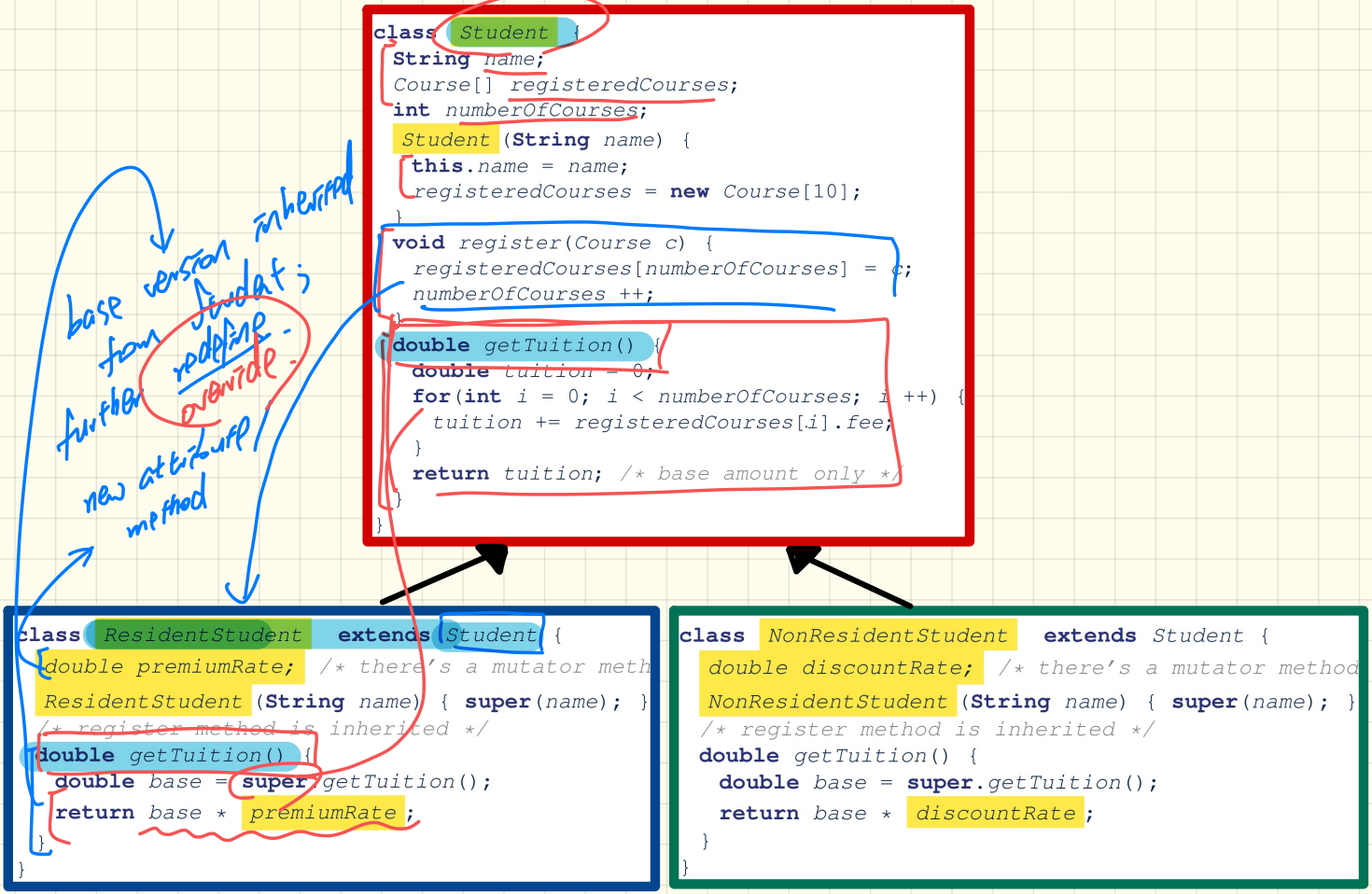
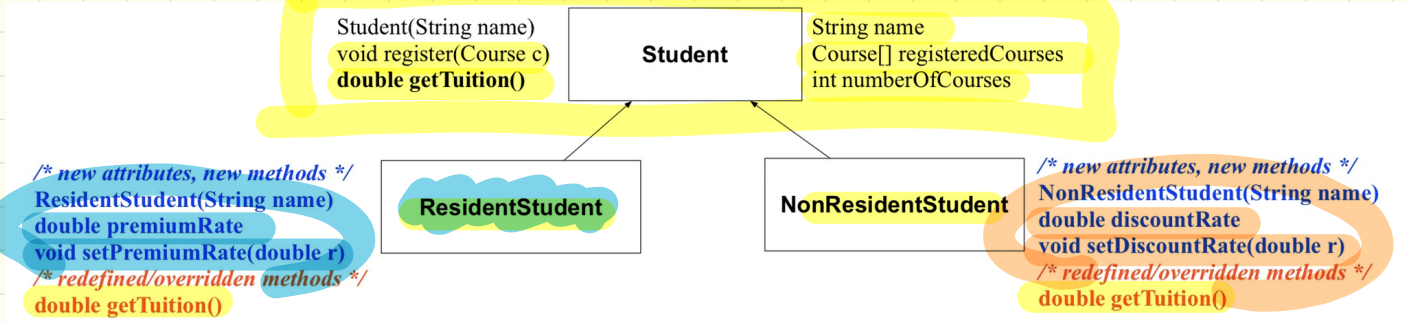
Lecture 17

Wednesday November 6

Review: Student Classes (with inheritance)



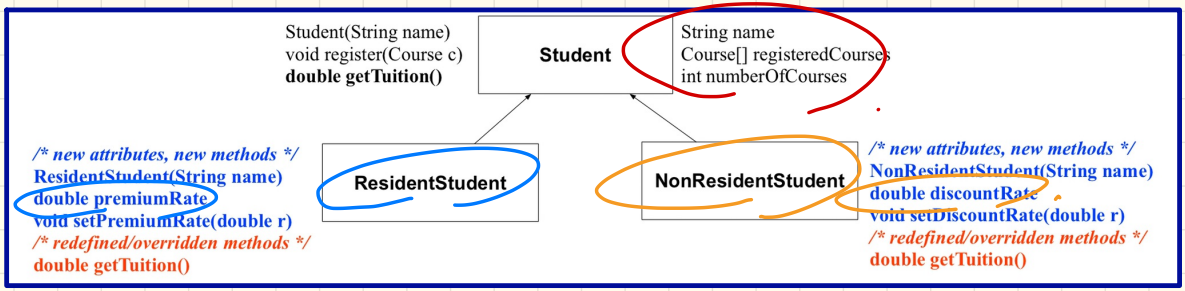
Review: Static Types and Expectations




```
Student s = new Student("Stella");
ResidentStudent rs = new ResidentStudent("Rachael");
NonResidentStudent nrs = new NonResidentStudent("Nancy");
```

	name	rcs	noc	reg	getT	pr	setPR	dr	setDR
s.									
rs.									
nrs.									

Review: Visualizing Parent and Child Objects



Inheritance Hirarchy

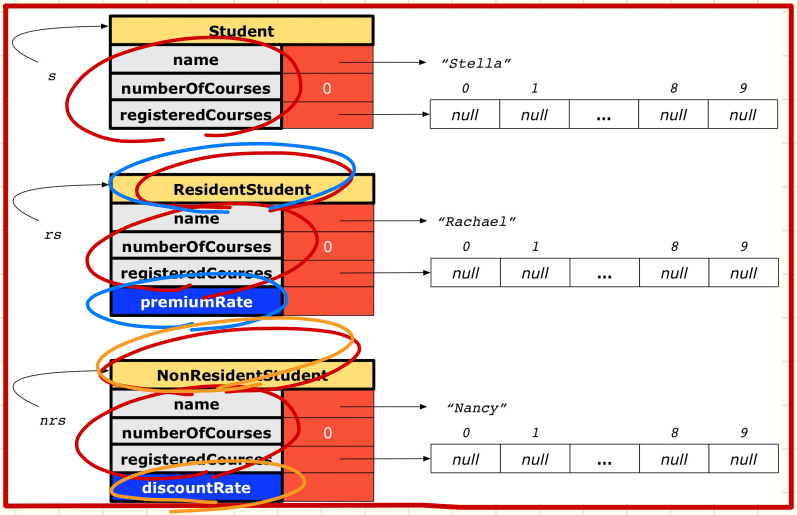
```
Student s = new Student("Stella");
ResidentStudent rs = new ResidentStudent("Rachael");
NonResidentStudent nrs = new NonResidentStudent("Nancy");
```

Declaring Static Types

static types

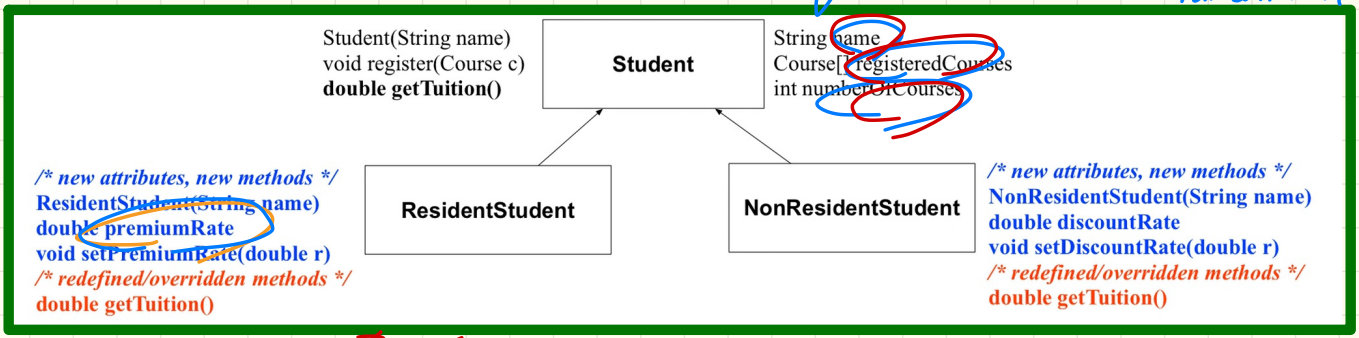
dynamic types

Runtime Object Structure



Intuition: Polymorphism

assignments in context of inheritance



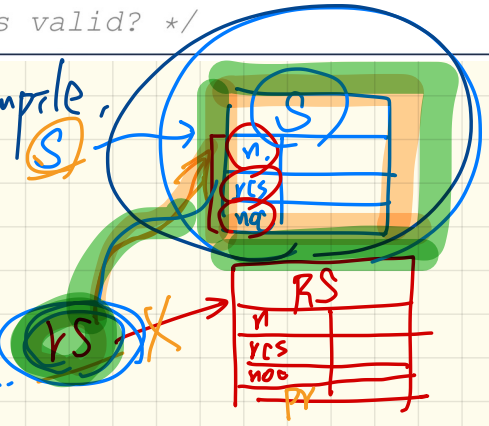

```
1  Student s = new Student("Stella");
2  ResidentStudent rs = new ResidentStudent("Rachael");
3  rs.setPremiumRate(1.25);
4  s = rs; /* Is this valid? */
5  rs = s; /* Is this valid? */
```

ST

Assume L5 should compile.

Execute rs = s;

Crash!! L5 should not compile.

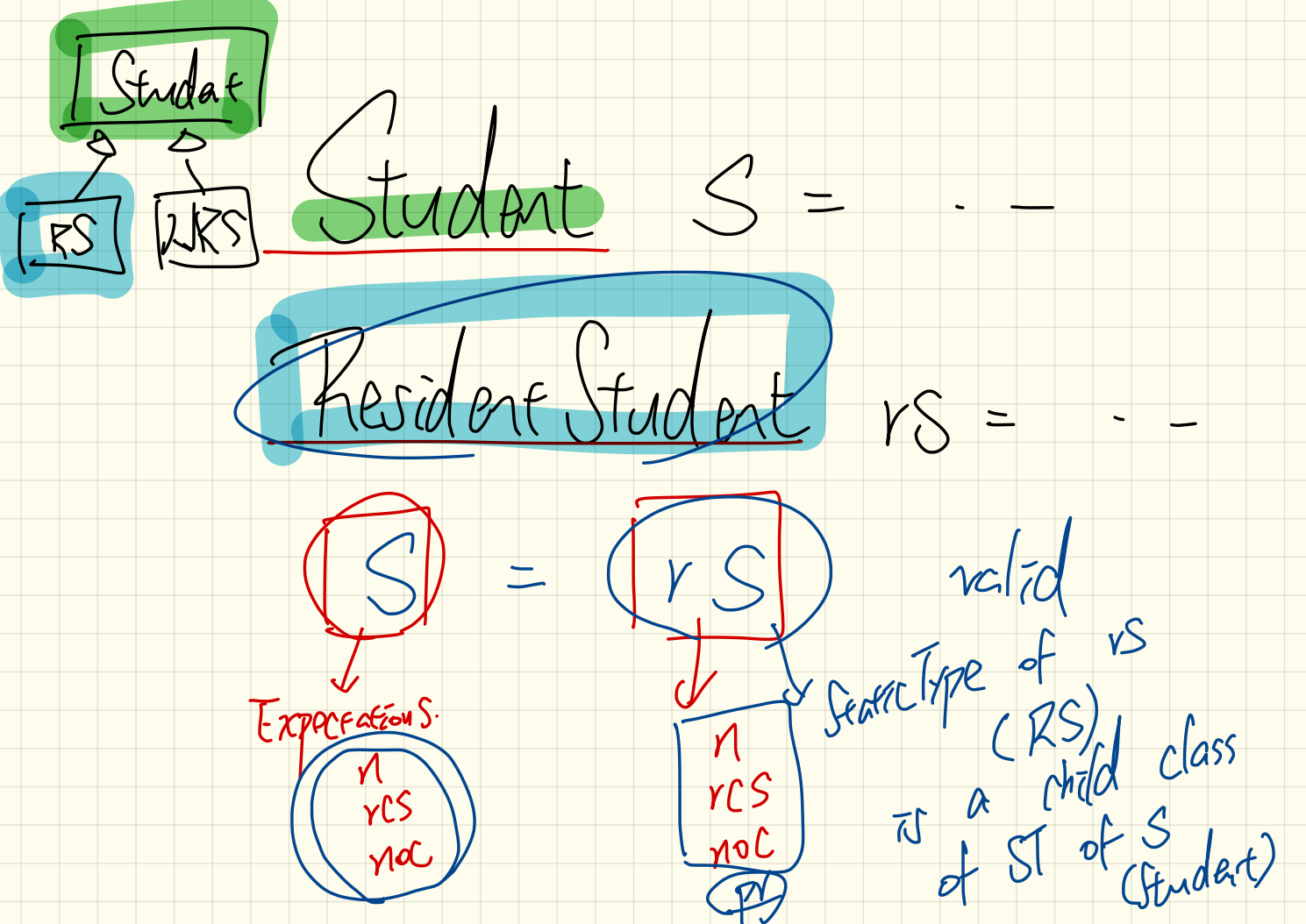


Expectations on rs.

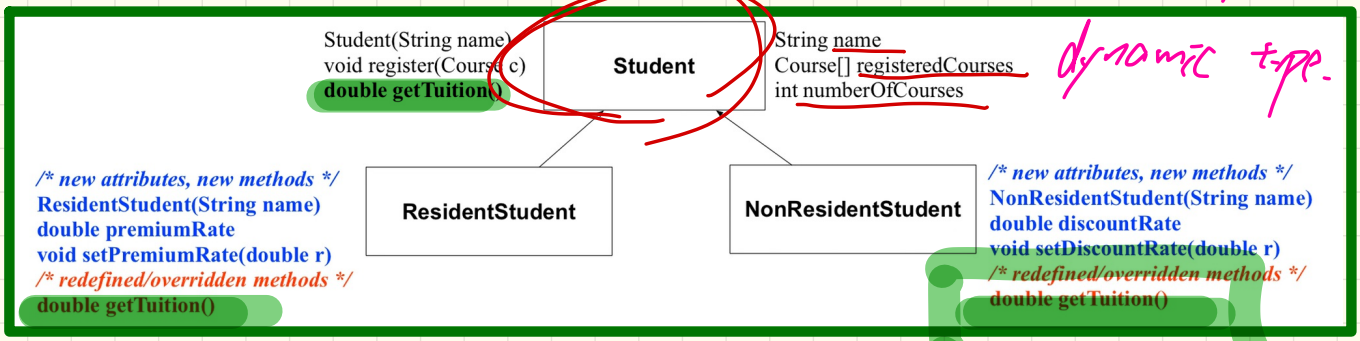
rs. n
rCS
noC

rs. pr

pr.


Studaf
RS
RRS
Student
S = . .
Resident Student
rS = . .
S
=
rS
valid
Expectations
n
rcs
noc
n
rcs
noc
PV
StaticType of rs
(RS)
is a child class
of ST of S
(Student)

Intuition: Dynamic Binding → runtime behaviour depends on dynamic type.




```
1  Course eecs2030 = new Course("EECS2030", 100.0);
2  Student s;
3  ResidentStudent rs = new ResidentStudent("Rachael");
4  NonResidentStudent nrs = new NonResidentStudent("Nancy");
5  rs.setPremiumRate(1.25); rs.register(eecs2030);
6  nrs.setDiscountRate(0.75); nrs.register(eecs2030);
7  s = rs; System.out.println(s.getTuition());/* output: 125.0 */
8  s = nrs; System.out.println(s.getTuition());/* output: 75.0 */
```

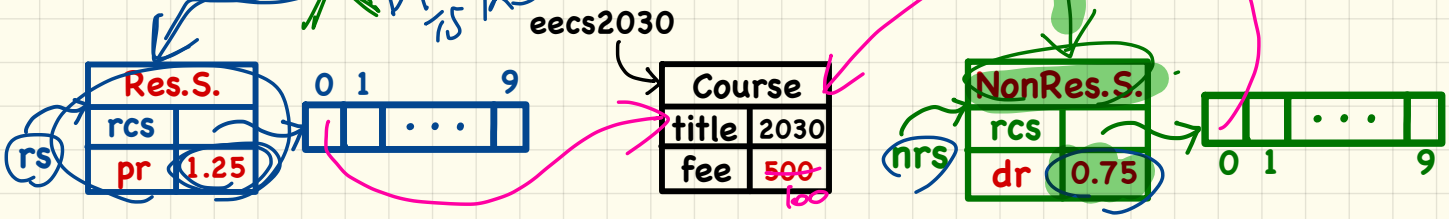
S.pr - X

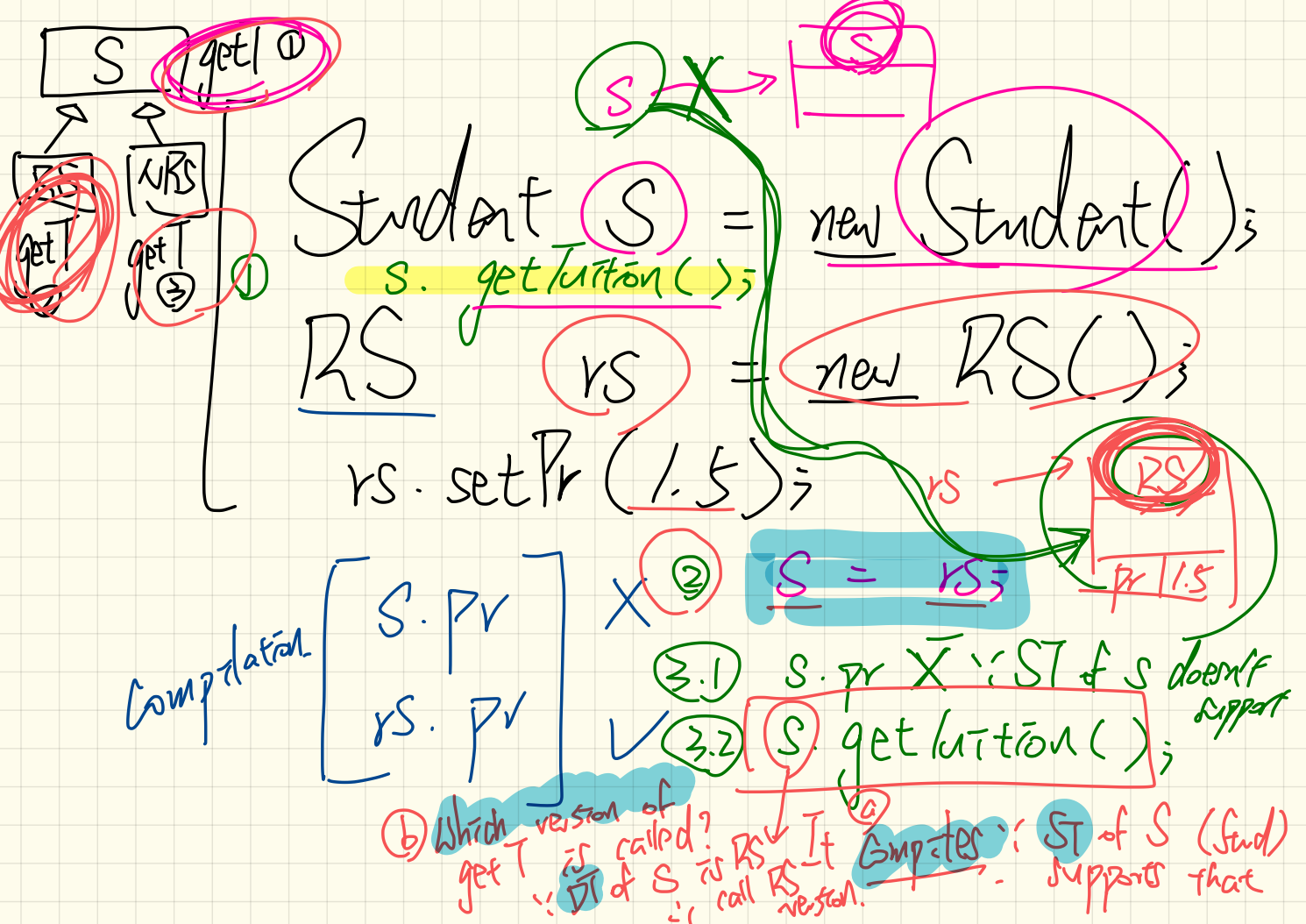
S: Student

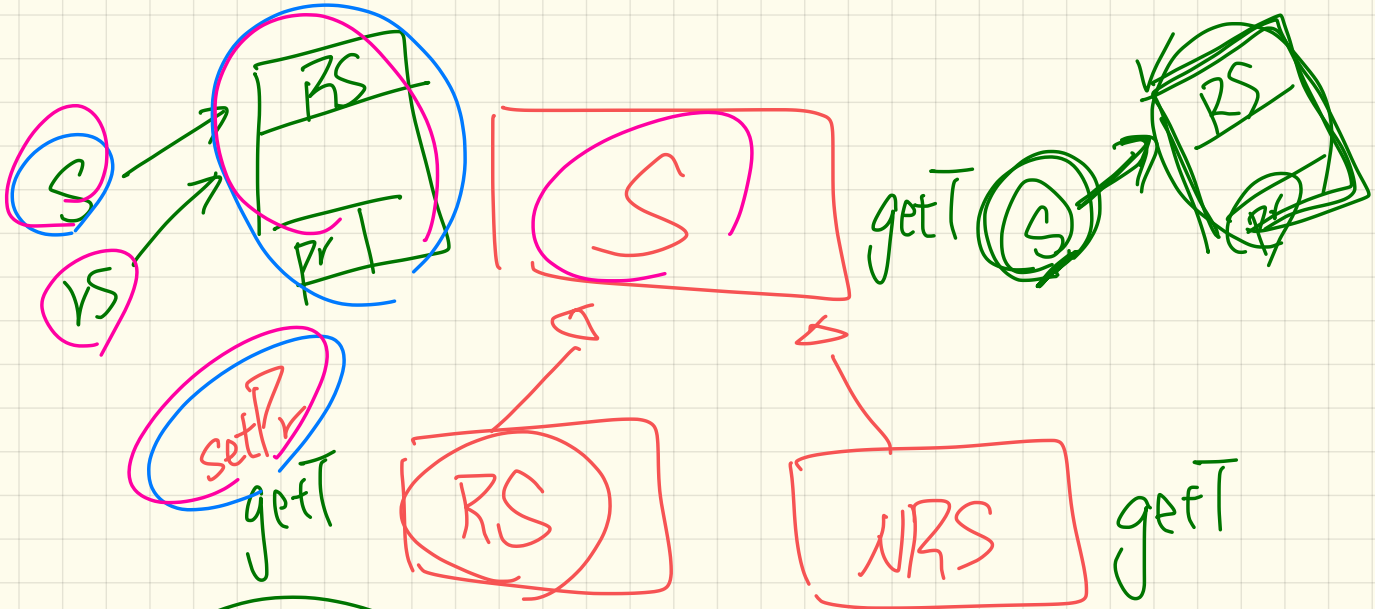
DT of S to NRS

DT of S is RS

S




S
getI ①
RS
getT ③
getT
①
Student S = new Student();
S. getTuition();
RS rs = new RS();
rs. setPr(1.5);
S
X
S
rs
RS
Pr 1.5
Compilation
S. Pr
rs. Pr
X ②
S = rs;
3.1 S. pr X ; ST of S doesn't support
3.2 S. getTuition();
⑥ Which version of getT is called?
∵ DT of S is RS ∴ call RS version.
④ It Compiles ∵ ST of S (Stud) supports that



Student S;

RS rS = new RS(..);

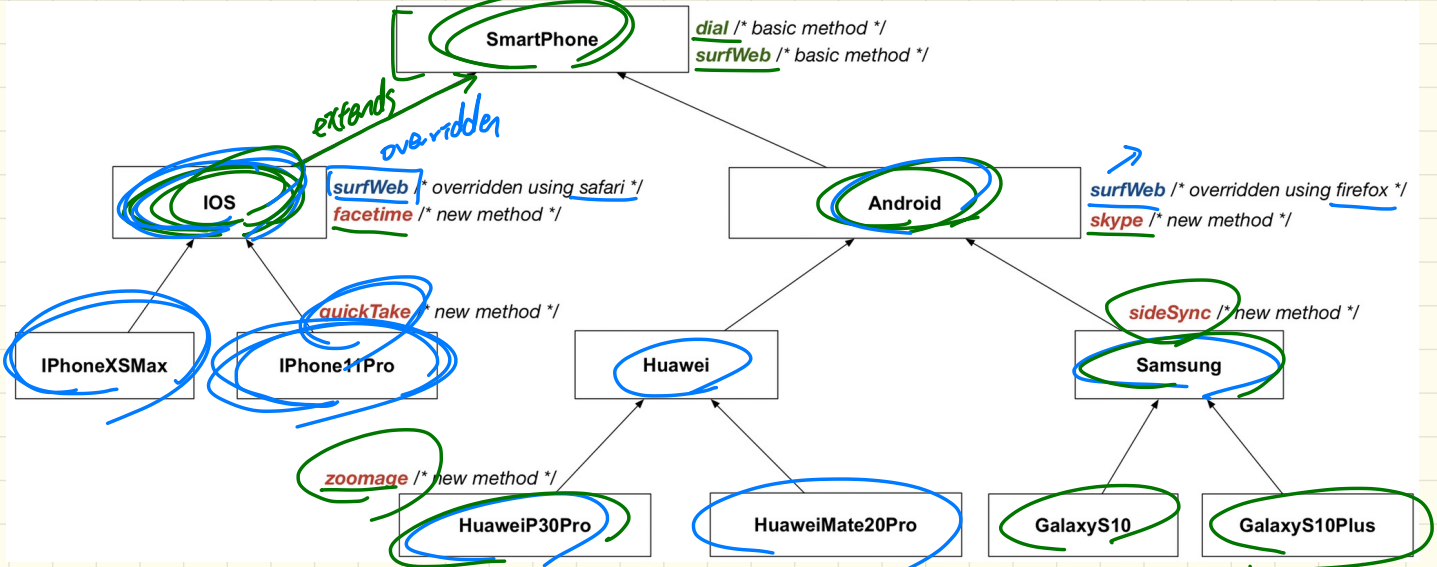
rS.setPr(..);

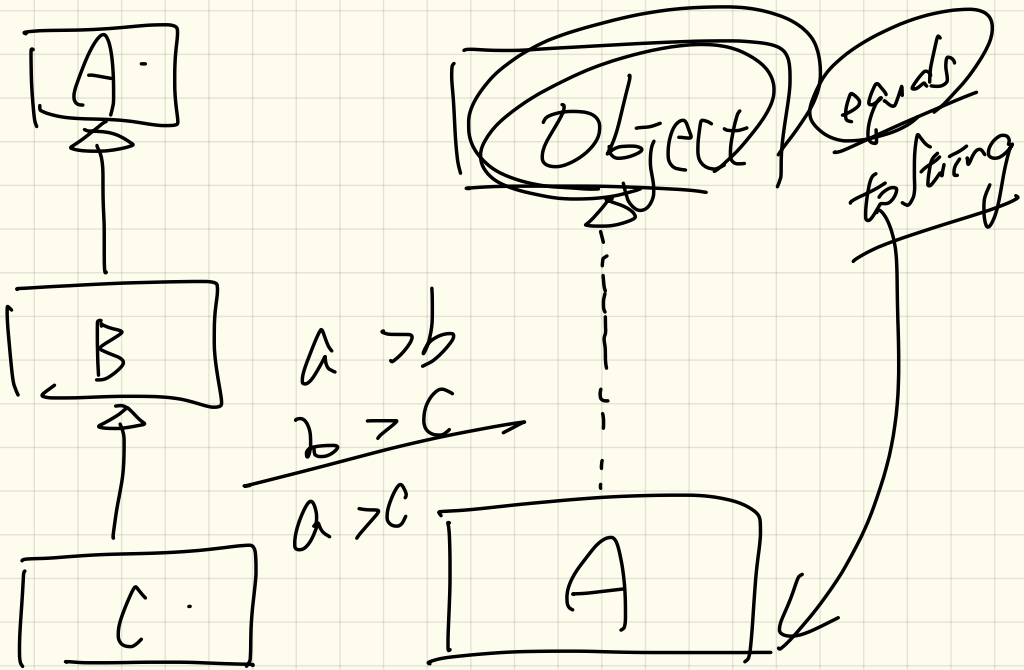
S = rS;

S setPr(1.5); ? X

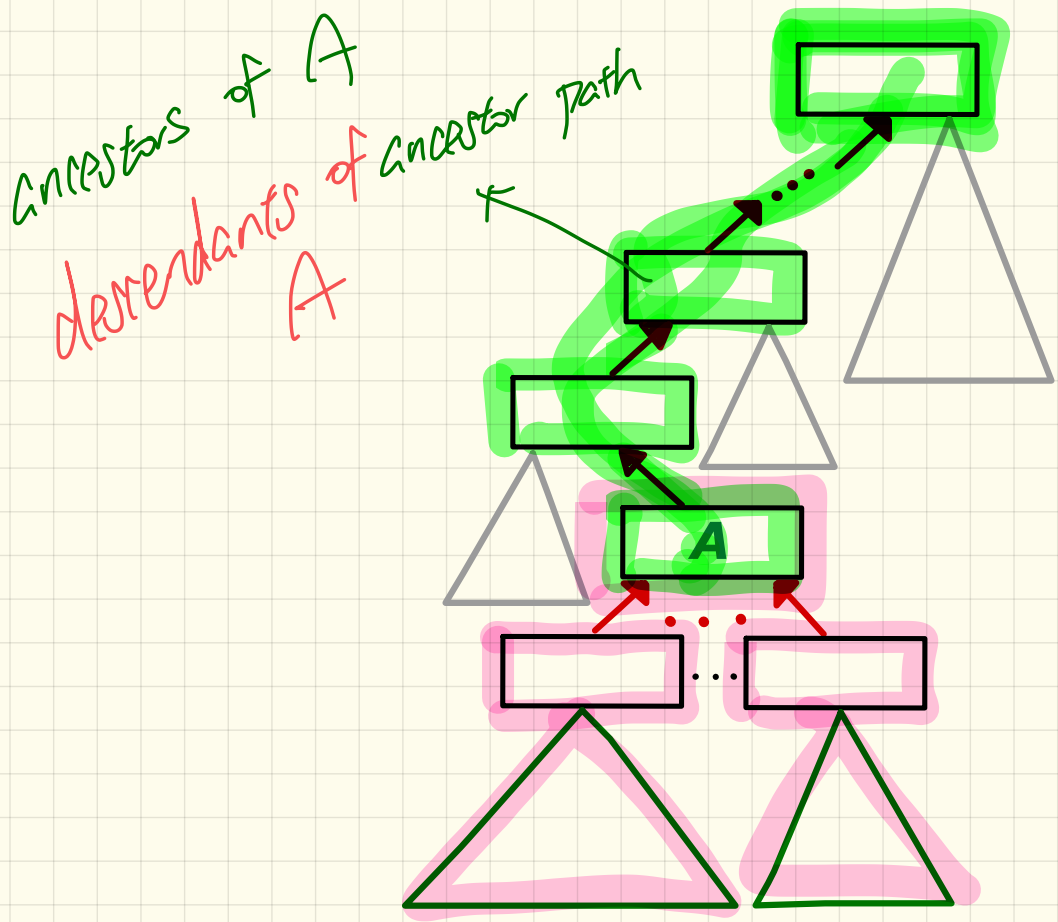
rS.setPr(2.7);

Multi-Level Inheritance Hierarchy of Smartphones



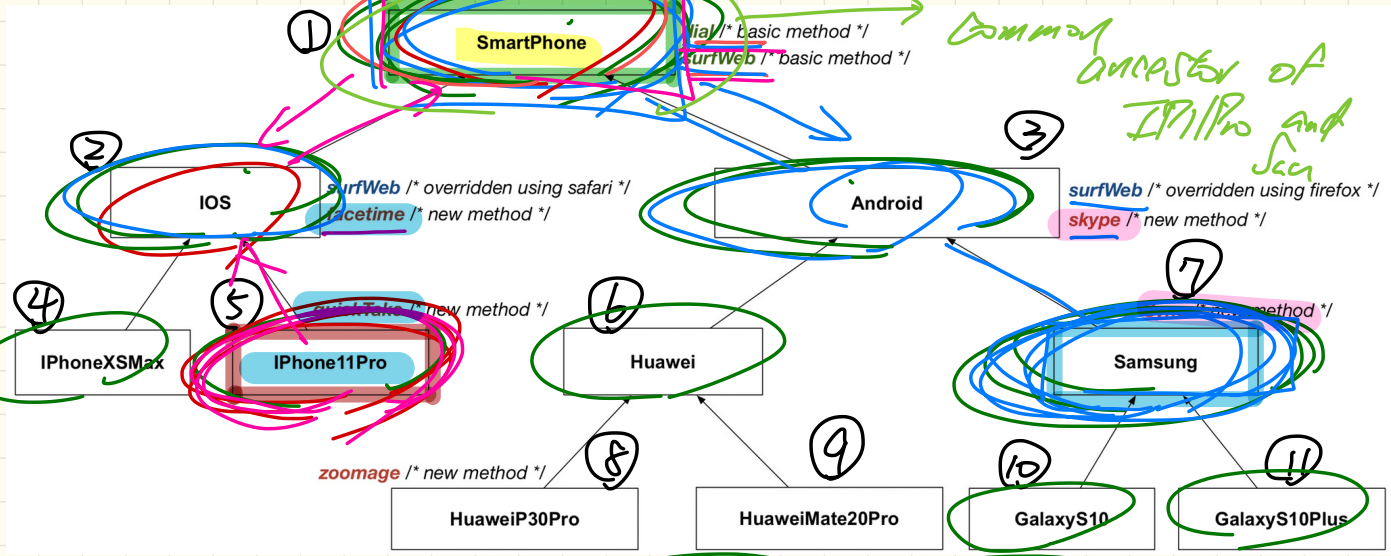

A
B
C
Object
equals
toString
a > b
b > c
a > c
A

Inheritance Forms a Type Hierarchy



Inheritance Accumulates Code for Reuse

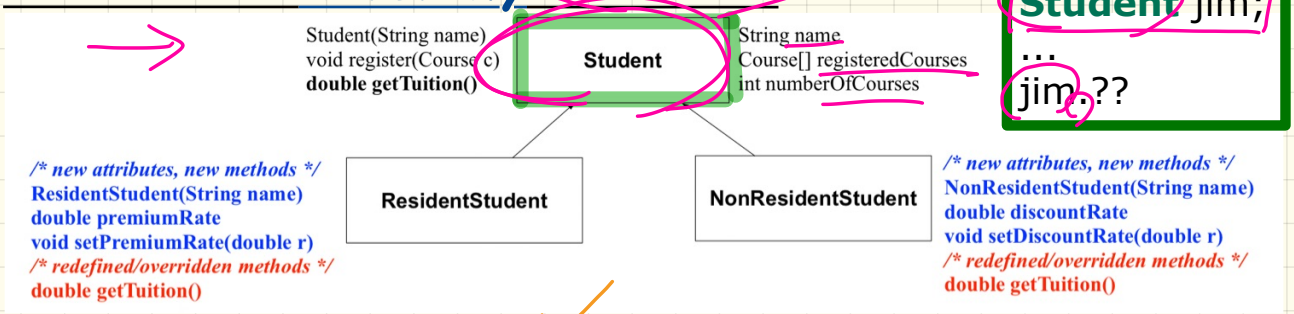
child classes



	ancestors	expectations	descendants
	SmartPhone	dial, surfWeb	① ~ ⑪.
	SanSung, Android SP	dial, surfWeb, skype, sideSync	⑦ ⑩ ⑪
	IP11Pro, IOS, SP	dial, surfWeb, facetime, guideTake	⑤.

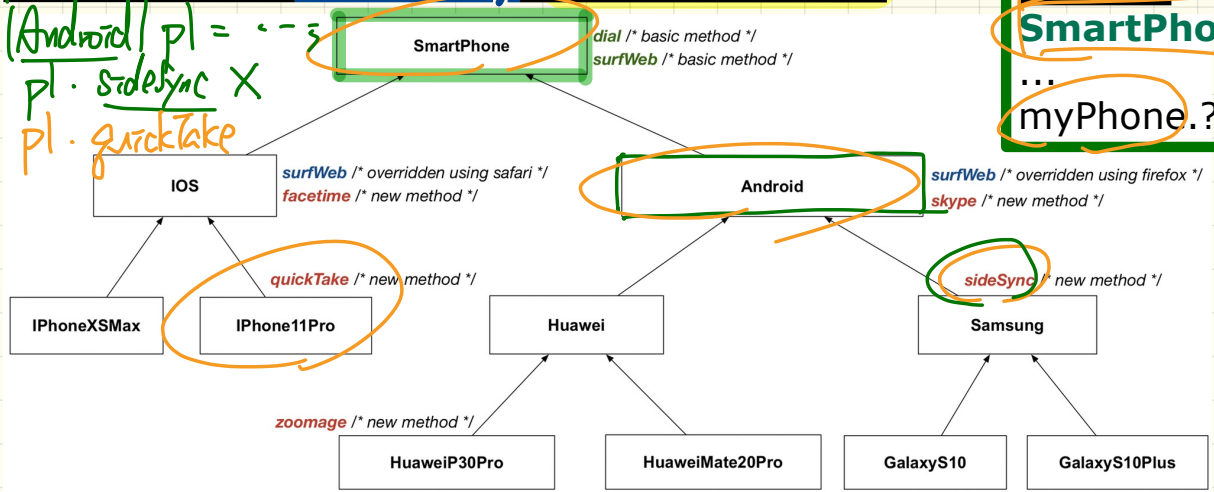
Static Types determine Expectations

Inheritance Herarchy: Students



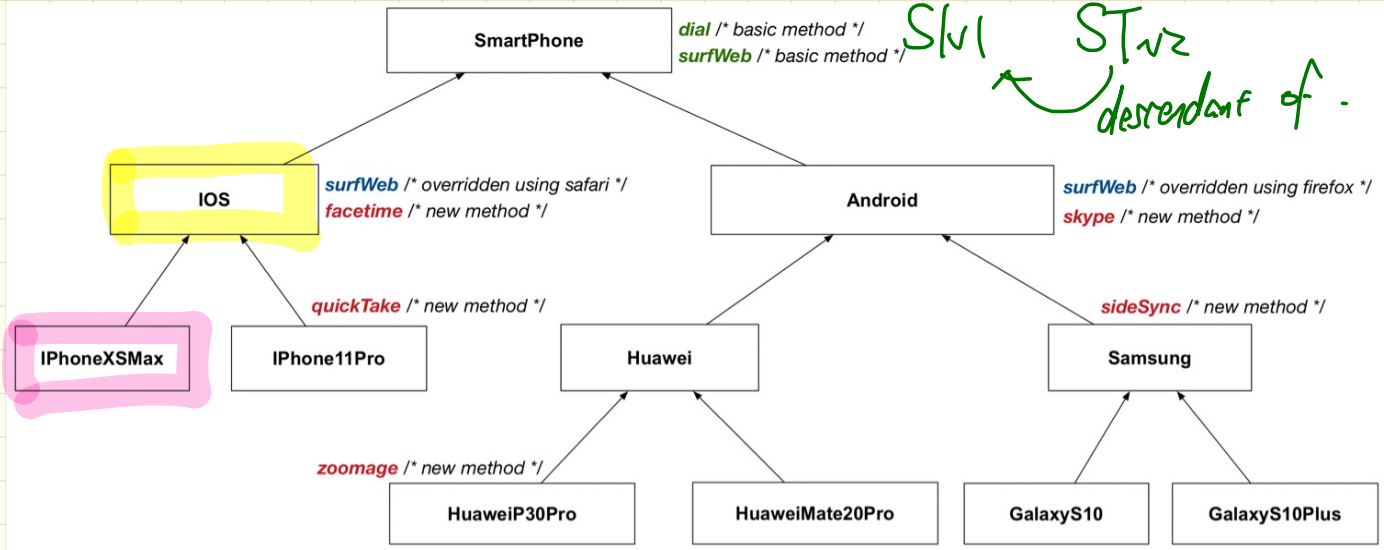
Declare:
Student jim;
...
jim.??

Inheritance Herarchy: Smart Phones



Declare:
SmartPhone myPhone;
...
myPhone.??

Rules of Substitutions (1)



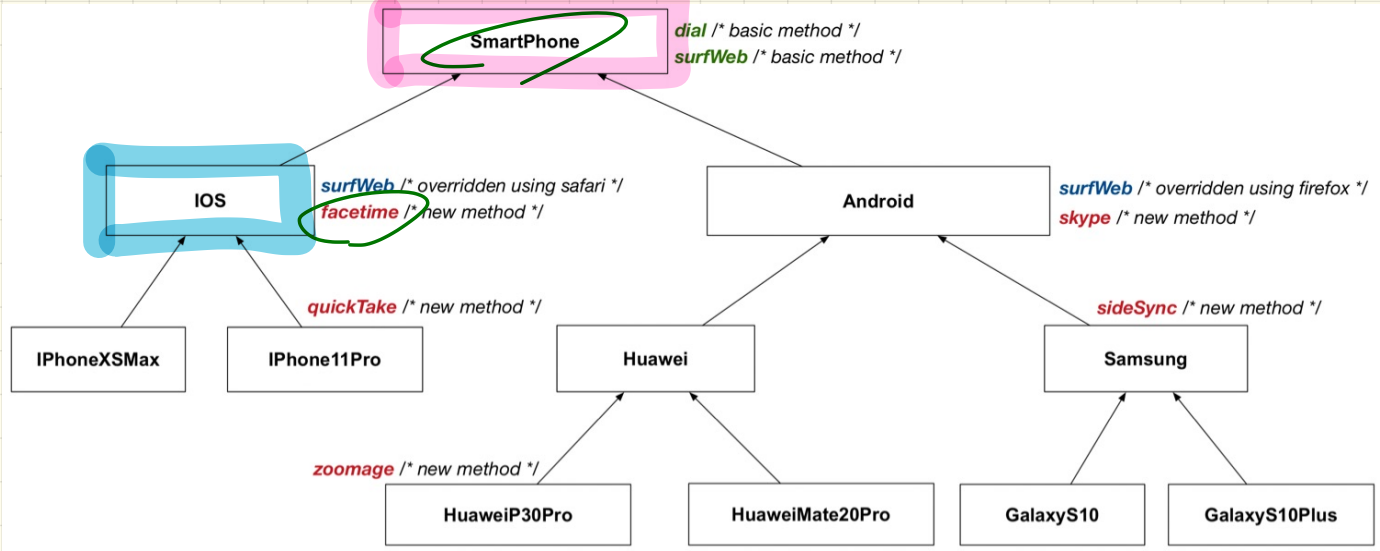
Declarations:

IOS sp1;
IPhoneXSMax sp2;
IPhonePro11 sp3;

Substitutions:

sp1 = sp2;
sp1 = sp3;

Rules of Substitutions (2)



Declarations:

IOS sp1;

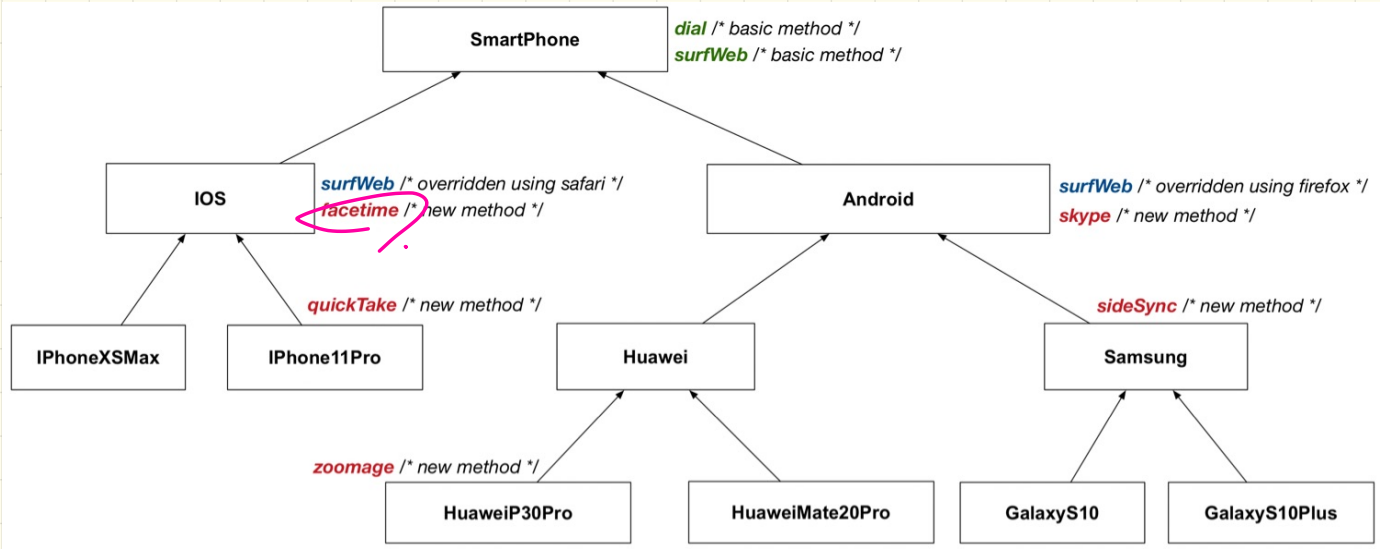
SmartPhone sp2;

Substitutions:

sp1 = sp2;

sp1: IOS sp2: SP

Rules of Substitutions (3)



Declarations:

IOS sp1;

HuaweiP30Pro sp2;

Substitutions:

sp1 = sp2;

e.g. facetime cannot be fulfilled by

ST: IOS

ST: HuaweiP30Pro

Visualization: Static Type vs. Dynamic Type

Declaration:

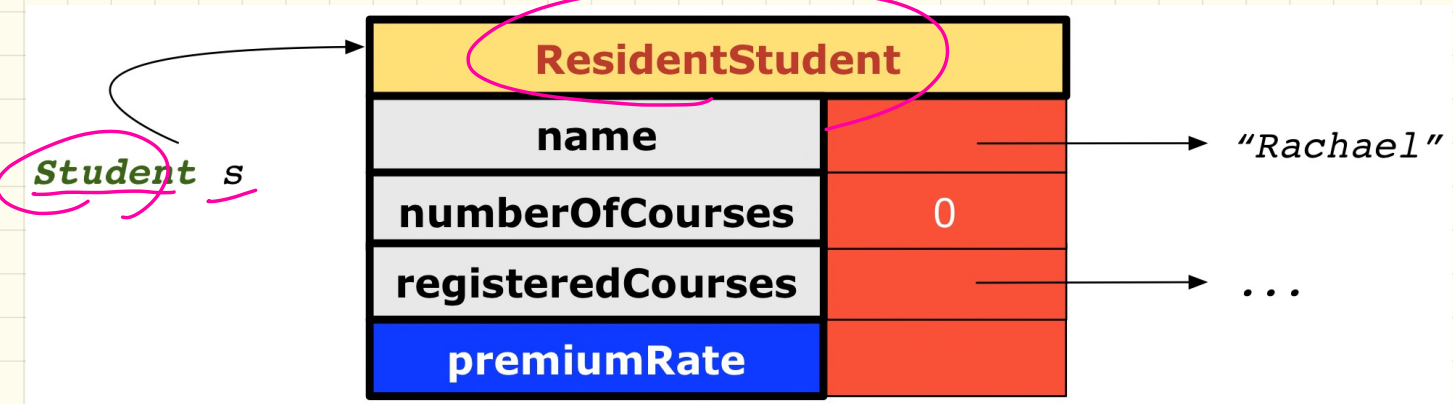
Student s;

ST

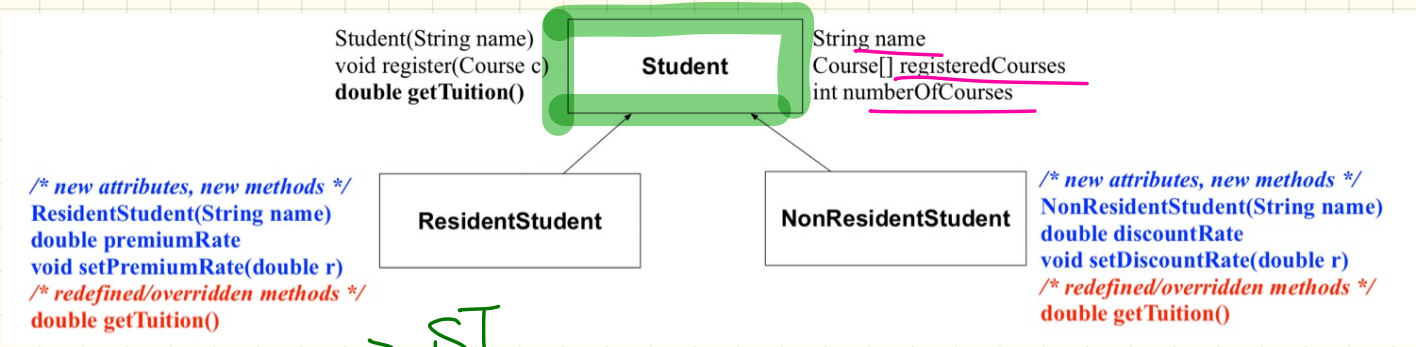
DT

Substitution:

s = **new ResidentStudent**("Rachael");



Change of Dynamic Type (1.1)



Example 1:

```
Student jim = new ResidentStudent(...);
jim = new NonResidentStudent(...);
```

ST

DT of jim: RS

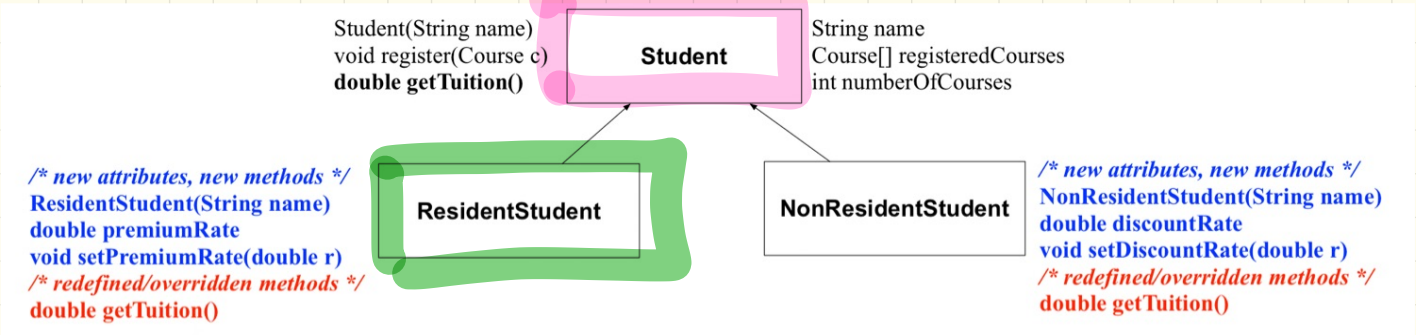
DT of jim: NRS

Student jim → RS / DV

jim → NRS / DV

can RS (the attempted DT) fulfil the expectation of jim (defined by ST Student)

Change of Dynamic Type (1.2)



Example 2:

```
ResidentStudent jeremy = new Student(...);
```

∴ Student cannot fulfill expectations of RS