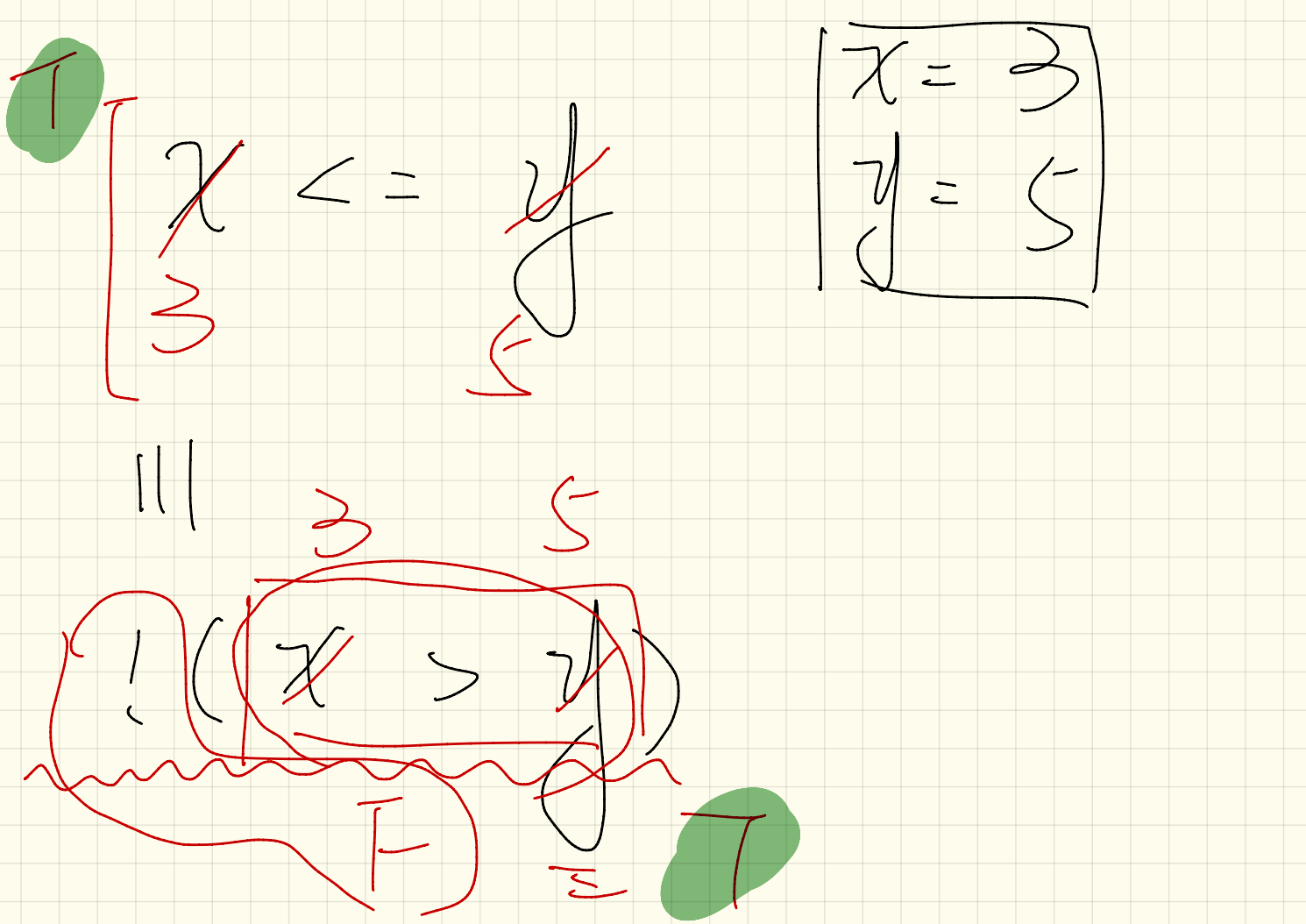
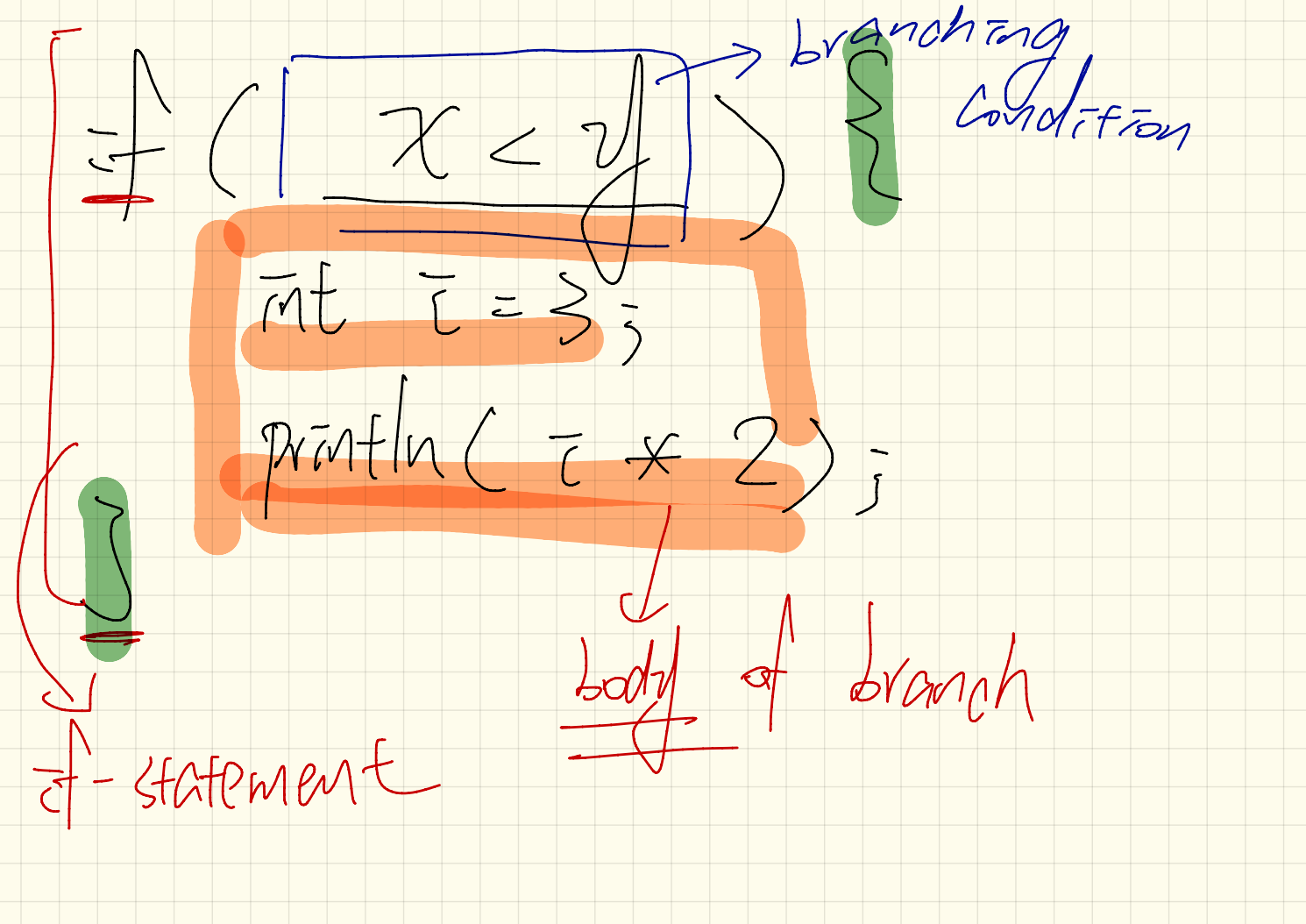
Monday Jan. 22

Lecture 3


T
x <= y
3
5
x = 3
y = 5
|||
3
5
! (x > y)
F
F
T


if (x < y)
branching condition
int i = 3;
println(i * 2);
body of branch
if-statement

if-statements

1

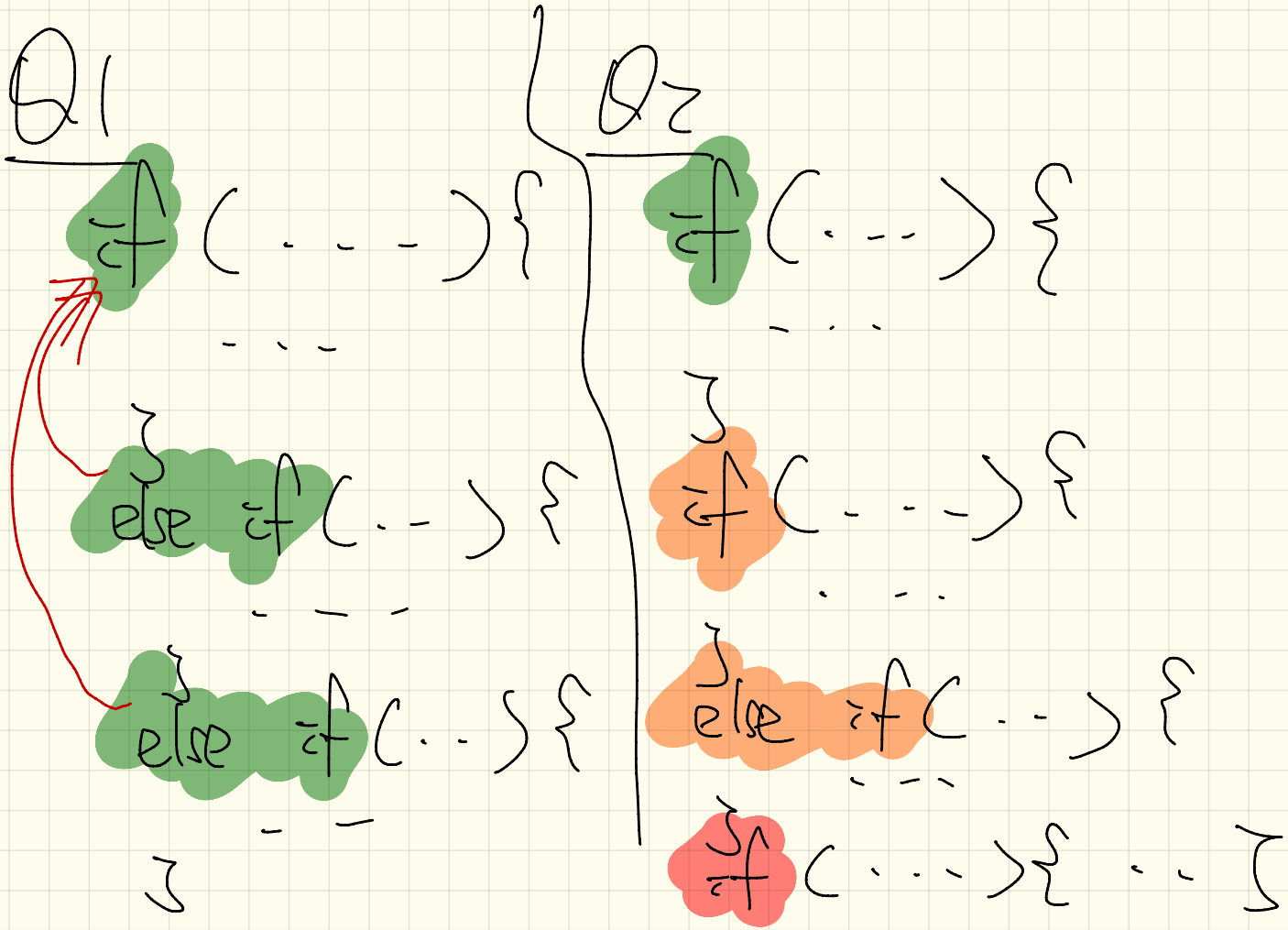
```
if ( ____ ) {
    [          ]
}
```

2

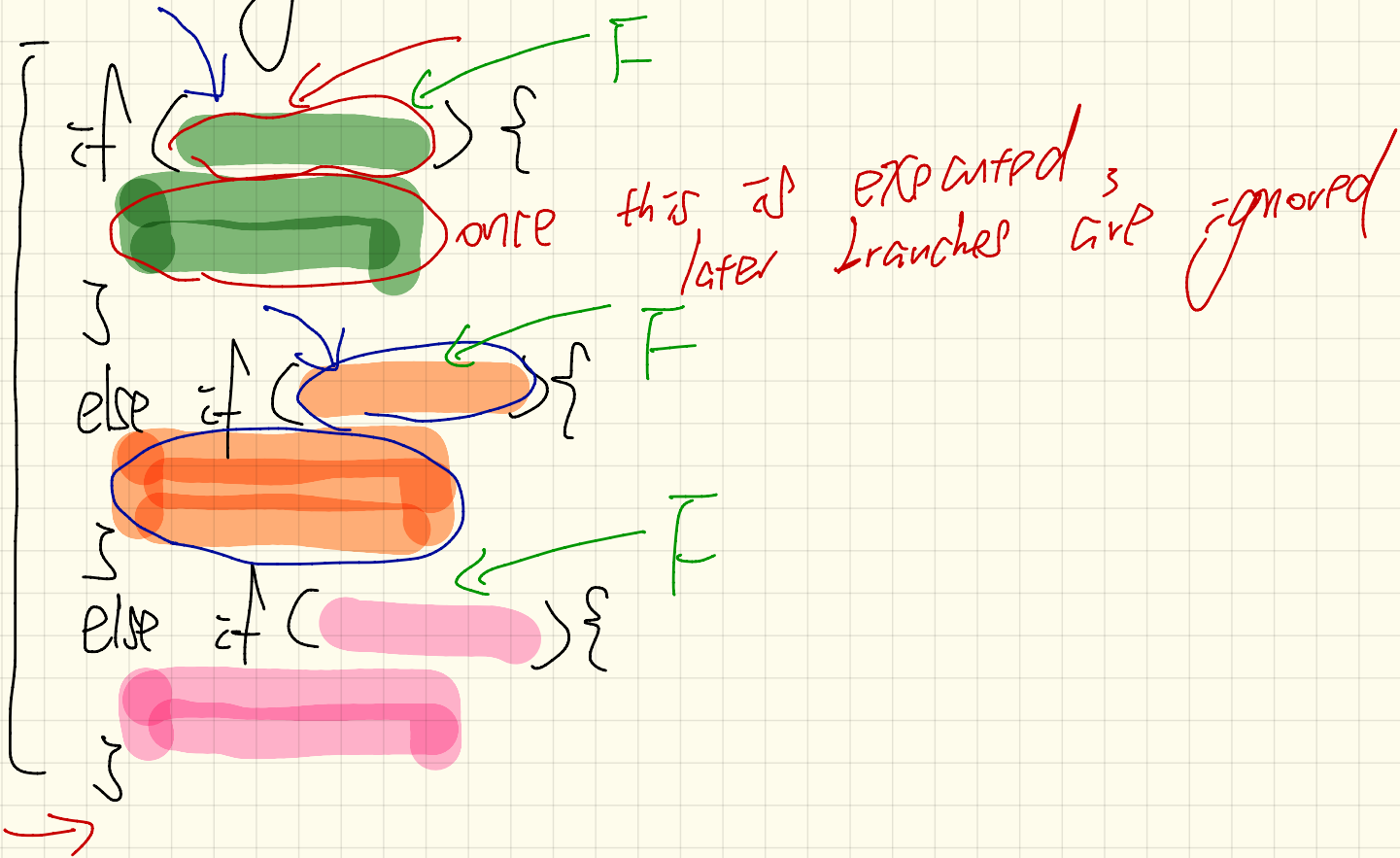
```
if ( ____ ) {
    [          ]
} else if ( ____ ) {
    [          ]
}
```

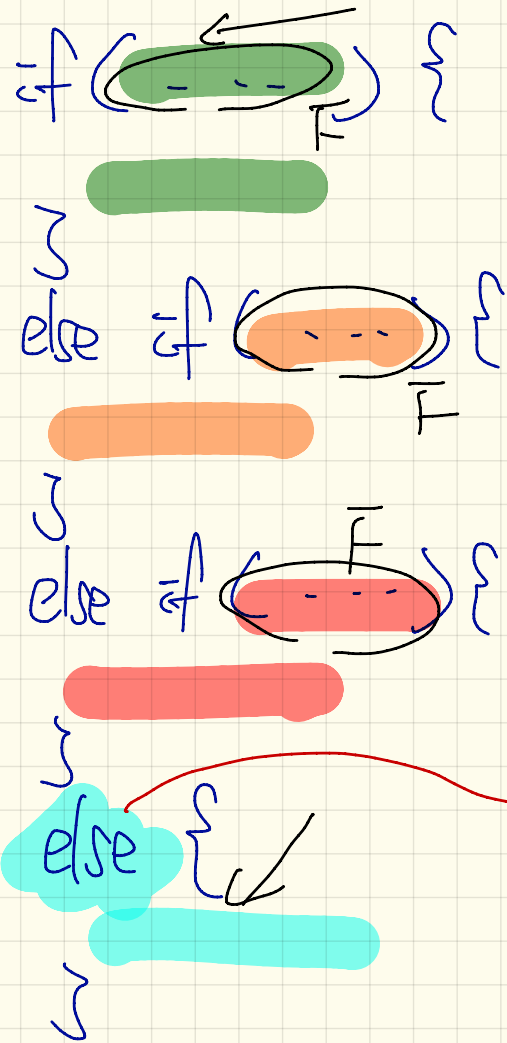
3

```
if ( ____ ) {
    [          ]
} else {
    [          ]
}
```


Q1
if (. . .) {
. . .
} else if (. .) {
. . .
} else if (. .) {
. . .
}
Q2
if (. . .) {
. . .
} if (. . .) {
. . .
} else if (. .) {
. . .
} if (. . .) { . . . }

Executing an if-statement





default branch:
when all previous branching conditions evaluate to F, this branch is executed.

```
int i = 5;
if(i < 0) {
  System.out.println("i is negative");
}
else if(i < 10) {
  System.out.println("i is less than than 10");
}
else if(i == 10) {
  System.out.println("i is equal to 10");
}
else {
  System.out.println("i is greater than 10");
}
```

i is less than 10

```
int i = 12;
if(i < 0) {
  System.out.println("i is negative");
}
else if(i < 10) {
  System.out.println("i is less than than 10");
}
else if(i == 10) {
  System.out.println("i is equal to 10");
}
```

12 < 0 F

12 < 10 F

12 == 10 F

```
int i = 12;
if(i < 0) {
  System.out.println("i is negative");
}
else if(i < 10) {
  System.out.println("i is less than than 10");
}
else if(i == 10) {
  System.out.println("i is equal to 10");
}
else {
  System.out.println("i is greater than 10");
}
```

12 < 0 F

12 < 10 F

12 == 10 F

radius < 0 invalid

valid ?

! (radius < 0)

radius >= 0

read input from user

① valid condition

```
if ( input is valid ){
    do comp.
}
else {
    print some error
}
```

error condition

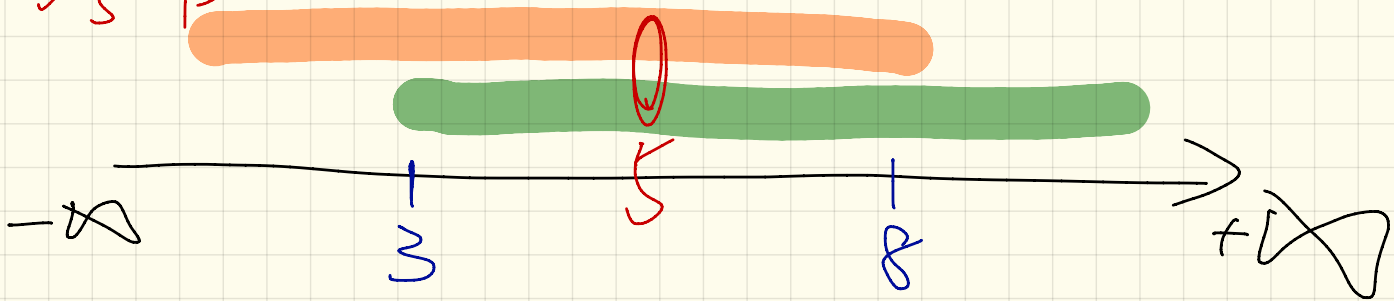
```
if ( input is invalid ){
    print some error
}
else {
    do comp.
}
```

1 if-statement

```
int i = 5;
if(i >= 3) {System.out.println("i is >= 3");}
else if(i <= 8) {System.out.println("i is <= 8");}
```

2 if-statements

```
int i = 5;
if(i >= 3) {System.out.println("i is >= 3");}
if(i <= 8) {System.out.println("i is <= 8");}
```

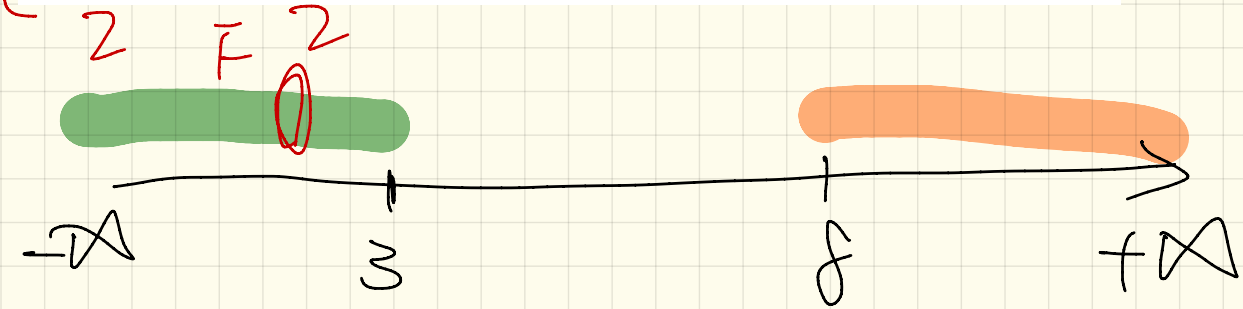


① ②

```
int i = 2;
if(i <= 3) {System.out.println("i is <= 3");}
else if(i >= 8) {System.out.println("i is >= 8");}
```

②

```
int i = 2;
if(i <= 3) {System.out.println("i is <= 3");}
if(i >= 8) {System.out.println("i is >= 8");}
```



Boolean Expressions

- true
  false
  ] Boolean literals

- x <= y
  y > z
  ] relational expressions

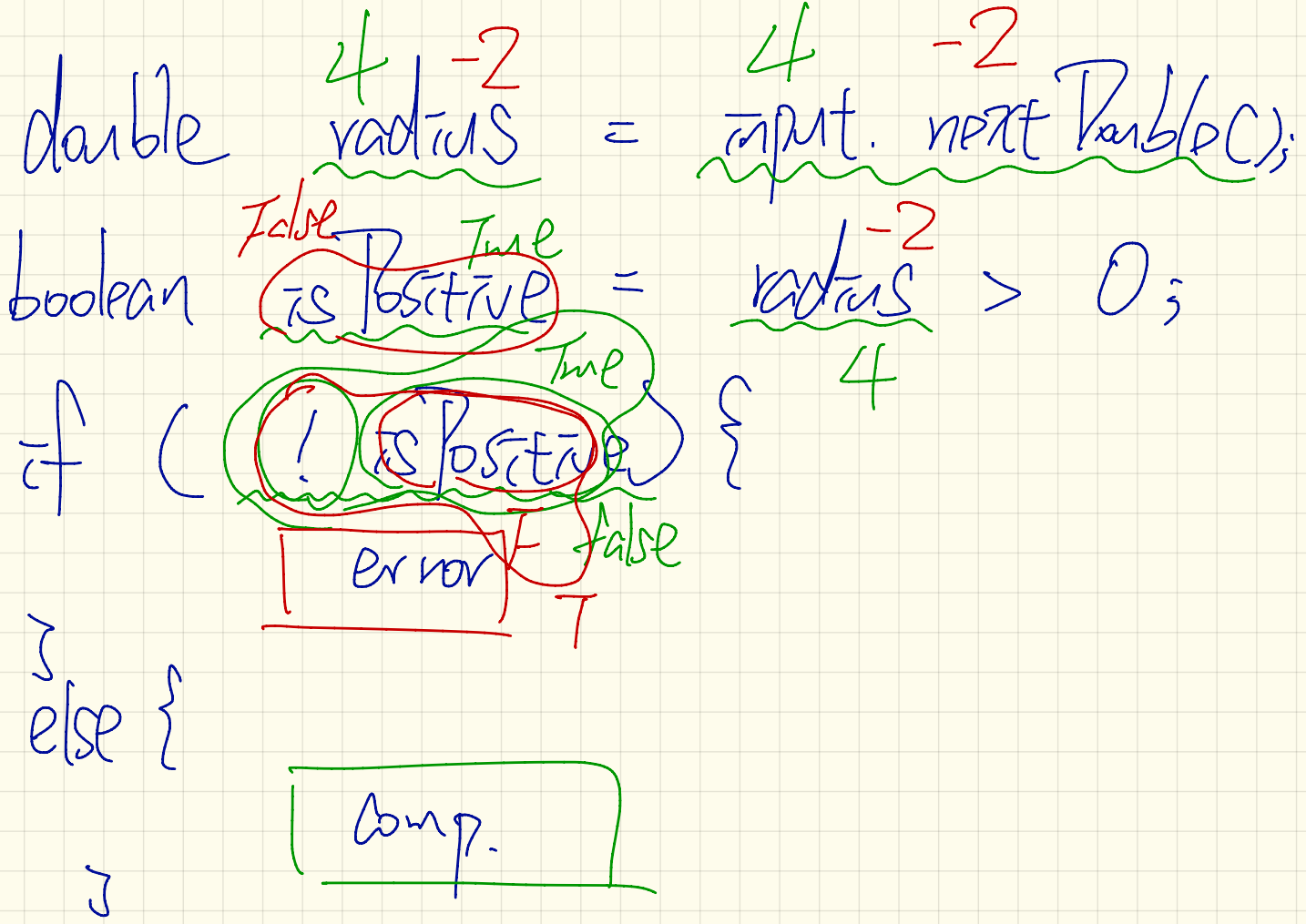
-] logical operations.

Logical negation (truth table)

unary operator

single operand

b	!b


4
-2
double radius = input.nextDouble();
boolean isPositive = radius > 0;
radius
4
false
true
true
if (! isPositive) {
error
false
}
else {
comp.
}

Conjunction

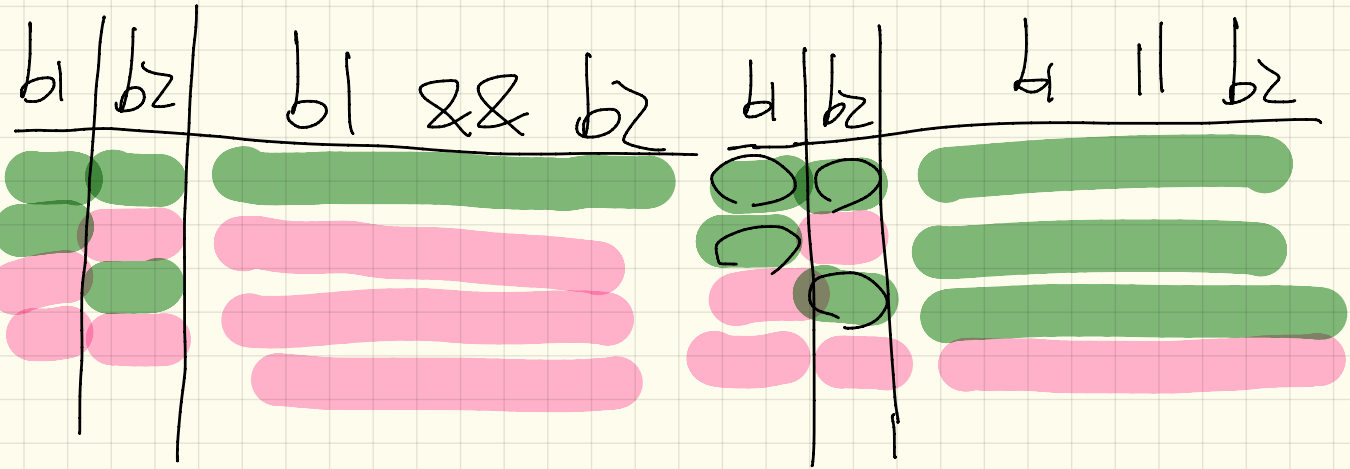
"and"

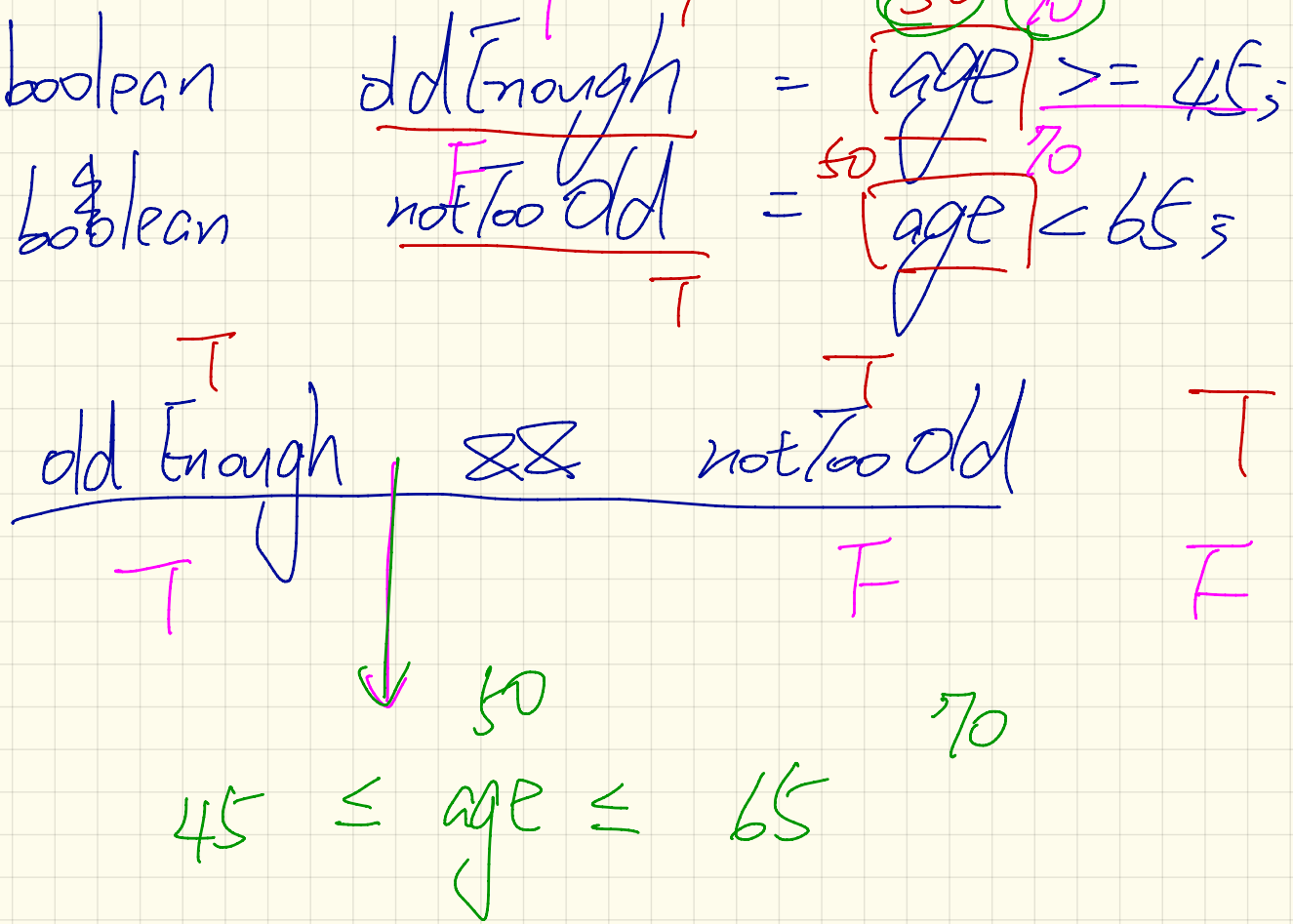
b1 && b2

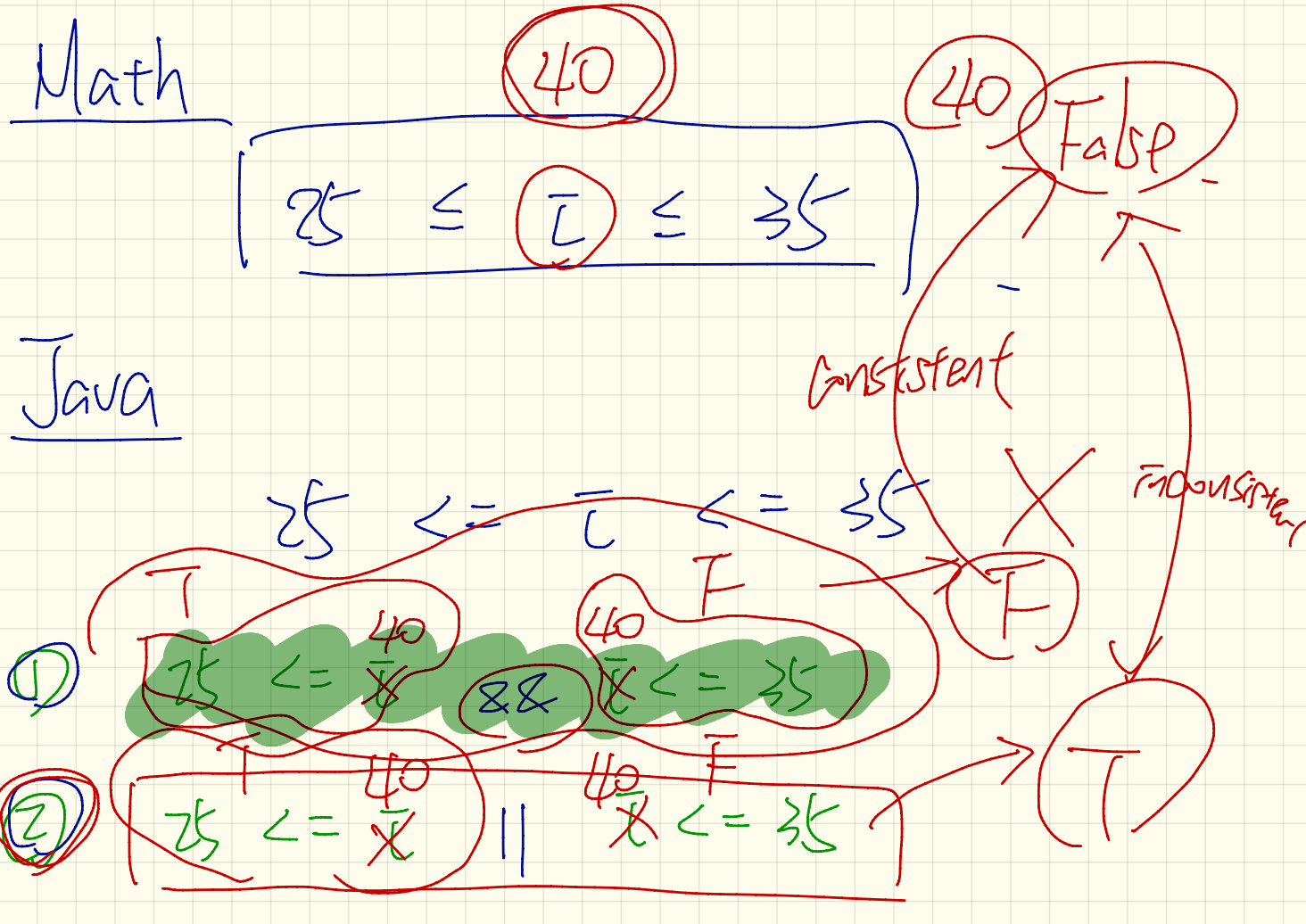
Disjunction

"or"

b1 || b2



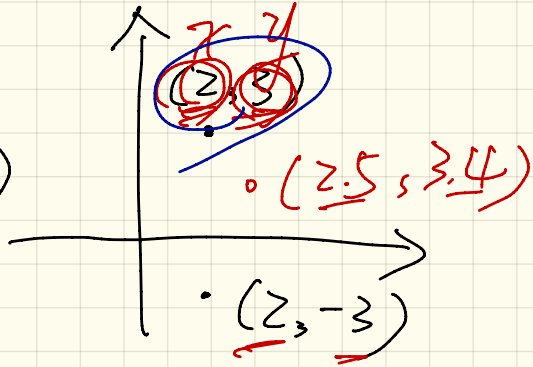

boolean oldEnough = age >= 45;
boolean notTooOld = age < 65;
oldEnough && notTooOld
45 ≤ age ≤ 65
50
70

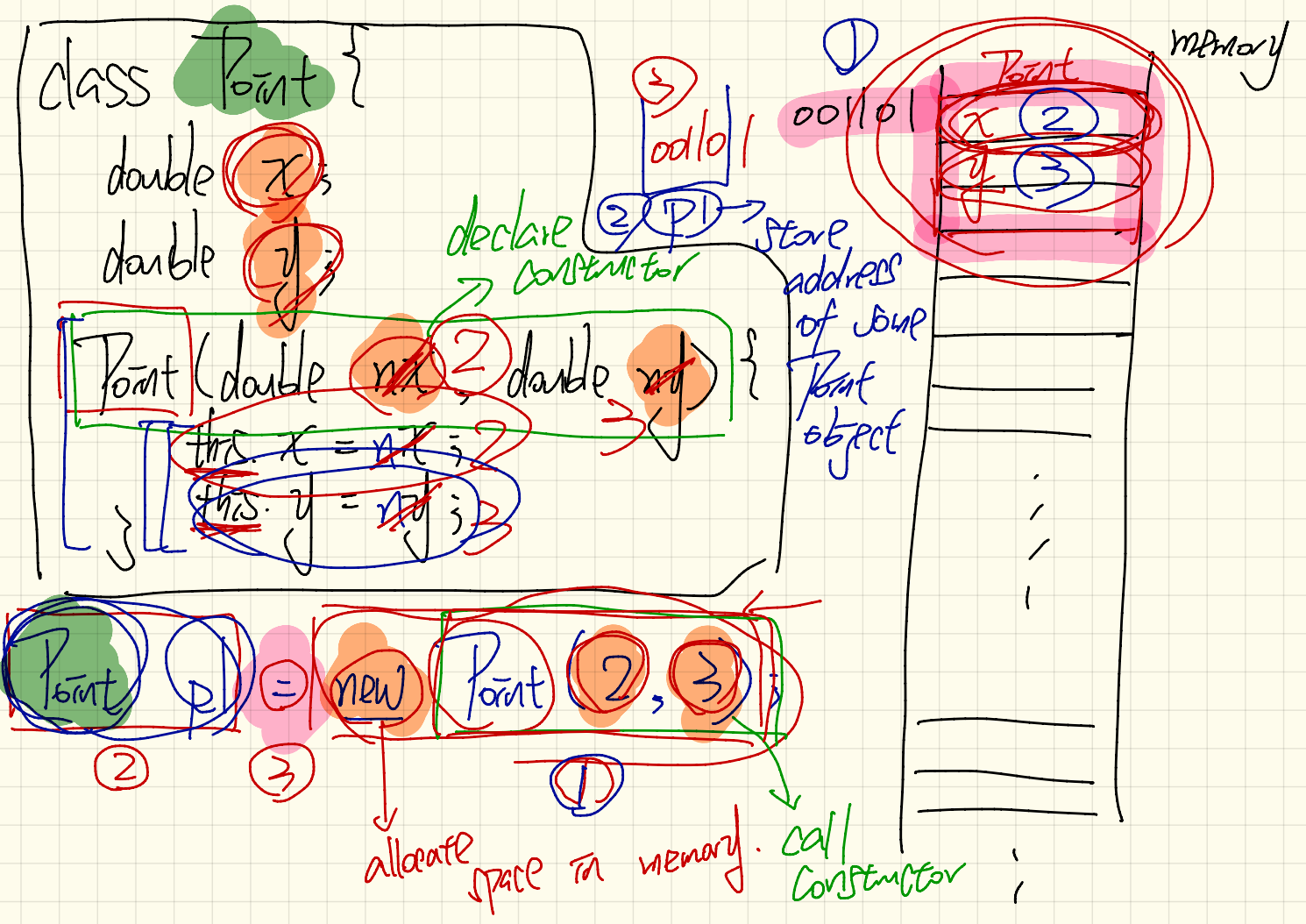

Math
40
25 ≤ L ≤ 35
Java
25 <= L <= 35
T
F
40
25 <= L
&&
L <= 35
①
②
||
40
False
Consistent
Inconsistent

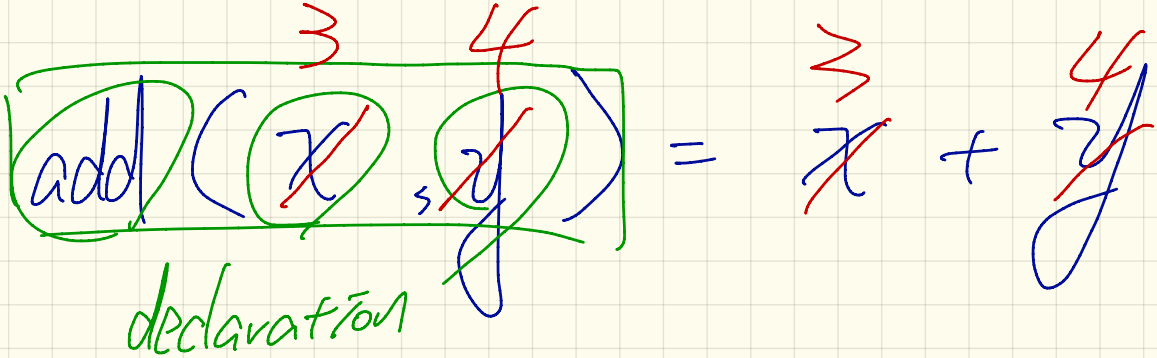
class Point {

- attributes (class variables)
- constructors
- accessor methods (questions)
- mutator methods (changes)

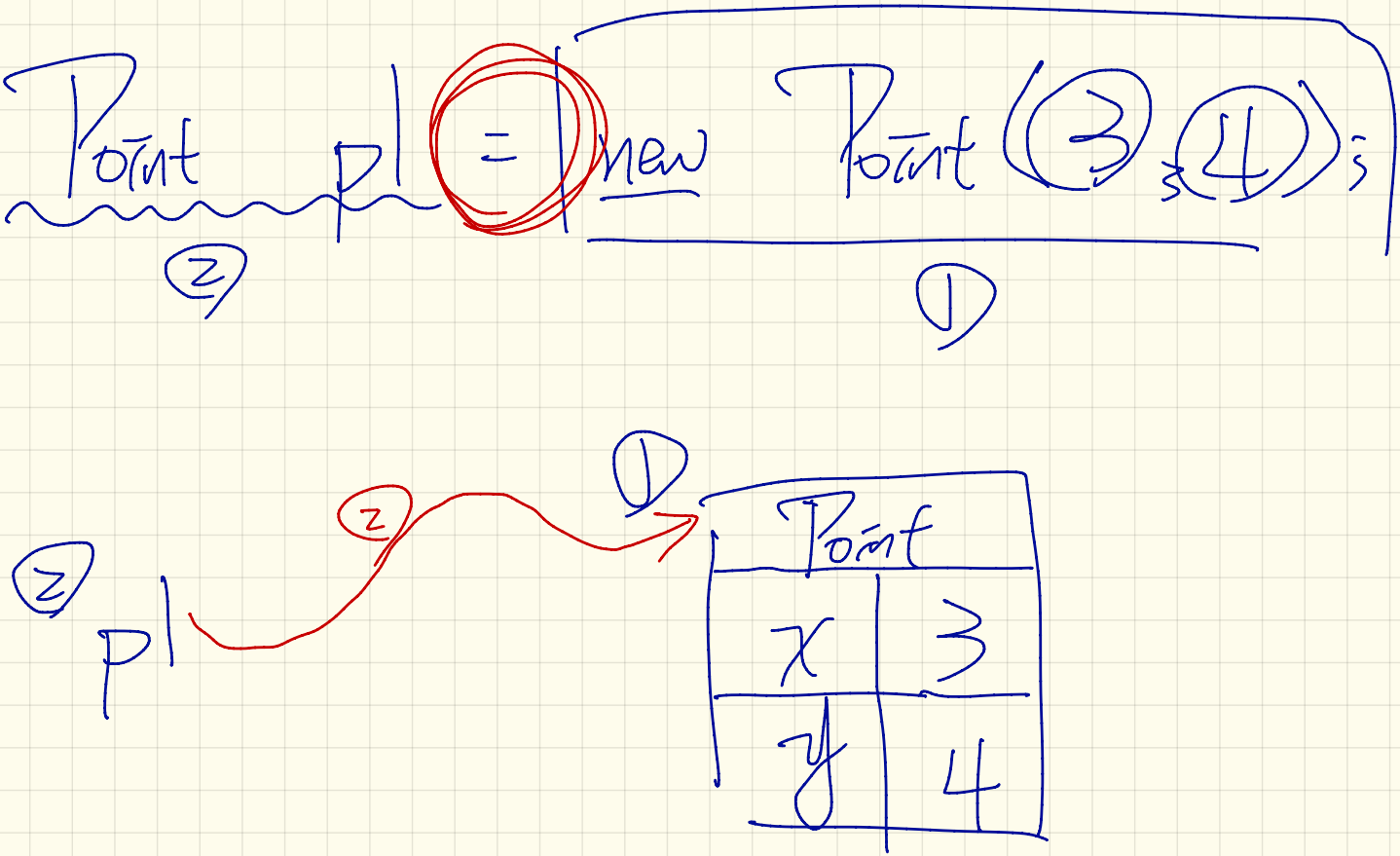
}




class Point {
double x;
double y;
Point(double x, double y) {
this.x = x;
this.y = y;
}
declare constructor
store address of some Point object
Point p1 = new Point(2, 3);
allocate space in memory.
call constructor
memory
Point
x 2
y 3



add (3 , 4) 7


Point p1 = new Point(3,4);
2
1
2
p1
2
1
Point
x
3
y
4