Wednesday March 27
Lecture 22

API: Math

utilities class

overloading → multiple methods have same name, but distinct param. types.

Modifier and Type	Method and Description
static double	**abs**(double a) · v1 — call the method using class name. — abs is an accessor — Math.abs(-2.8) → 2.8 (param.) Returns the absolute value of a double value.
static float	**abs**(float a) v2 Returns the absolute value of a float value.
static int	**abs**(int a) v3 Returns the absolute value of an int value.
static long	**abs**(long a) v4 — identical method name Returns the absolute value of a long value.

return type.

Definition

API of Math

static double abs (double x) ← parameter

Usage

Math. abs (-2.4) ← argument .

Math. random() → 0

① ✗
(int) (Math. random() * 100
[0, 1.0)
0.79
0

② (int) (Math.random() * 100)
[0, 1.0) 0.79 * 100
79

79

genetics

0
⋮
1

(int) (Math. random () *100)
0.23333
23.333 ✗

[0, 1.0)   23

[0.2] → 20

0.79 → 79

0.99

0.23333

1.0 ✗
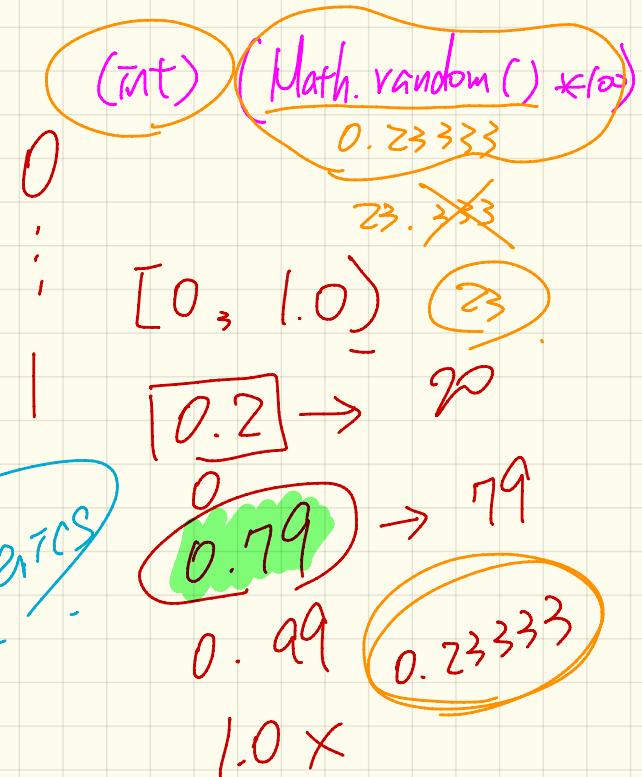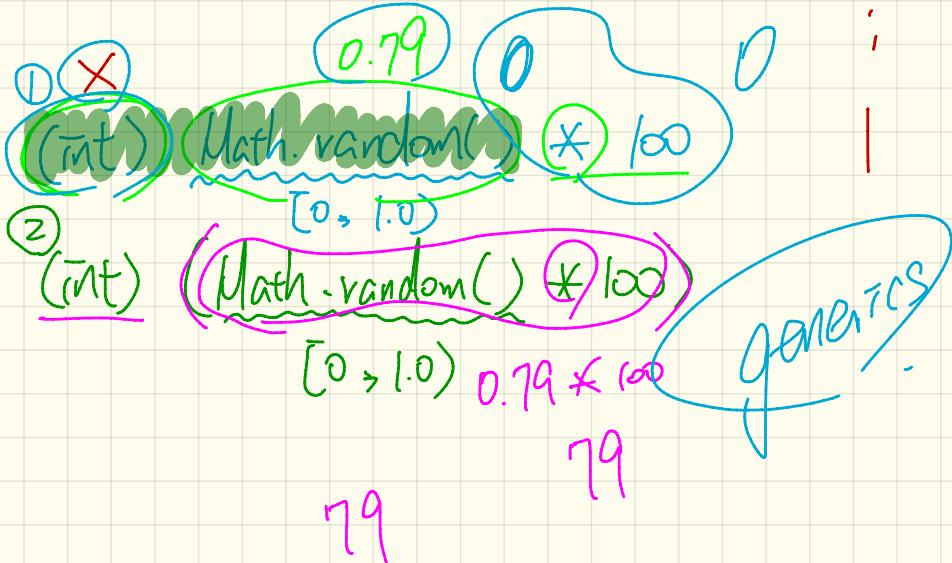
non-static methods

API: ArrayList

int	**size**() Returns the number of elements in this list.
boolean	**add**(E e) → element type Appends the specified element to the end of this list.
void	**add**(int index, E element) Inserts the specified element at the specified position in this list.
boolean	**contains**(Object o) Returns true if this list contains the specified element.
E	**remove**(int index) Removes the element at the specified position in this list.
boolean	**remove**(Object o) Removes the first occurrence of the specified element from this list, if it is present.
int	**indexOf**(Object o) Returns the index of the first occurrence of the specified element in this list, or -1 if this list does not contain the element.
E	**get**(int index) Returns the element at the specified position in this list.

ArrayList <String>
list1;

ArrayList <Person>
list2;

ArrayList < String > list = new . --

list . size()

(list) . length ✗ does not compile
∴ length is not part of
the API of ArrayList

Use of ArrayList

list →

```
ArrayList<String> list = new ArrayList<String>();
println(list.size());
println(list.contains("A"));
println(list.indexOf("A"));
list.add("A");
list.add("B");
println(list.contains("A")); println(list.contains("B")); println(list.contains("C"));
println(list.indexOf("A")); println(list.indexOf("B")); println(list.indexOf("C"));
list.add(1, "C");
println(list.contains("A")); println(list.contains("B")); println(list.contains("C"));
println(list.indexOf("A")); println(list.indexOf("B")); println(list.indexOf("C"));
list.remove("C");
println(list.contains("A")); println(list.contains("B")); println(list.contains("C"));
println(list.indexOf("A")); println(list.indexOf("B")); println(list.indexOf("C"));

for(int i = 0; i < list.size(); i ++) {
  println(list.get(i));
}
```