Monday March 5

Lecture 8

Input: array of elements

A

int[]

positive

String[]

equal to "A"

witness of violation

Problem: (1) Are all elements in A satisfied with a property. (return false if a wit. of vio. is found)

(2) Is there at least an element in A satisfied with a property. (return true if a wit. of satisf. is found)

witness of satisfaction

Input: empty array.

(1) true (∵ no wit. of vio.)

(2) false (∵ no wit. of satisf.).

(1) $\forall x \mid x \in \phi \cdot P(x)$] tmp

False

(2) $\exists x \mid x \in \phi \cdot P(x)$] false

False

(1) all numbers positive

```
boolean soFar = true;
for(int i=0; i< a.length; i++){
    soFar = soFar && a[i] > 0;
}
```

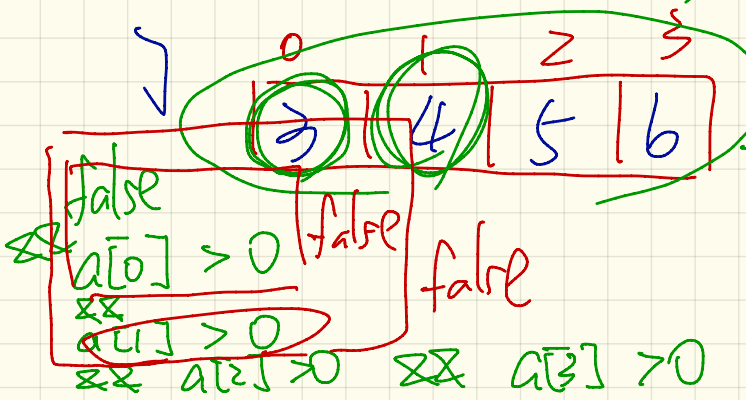
(2) at least one number is positive

(1) all numbers positive

boolean soFar = true;

for (int i=0; i< a.length; i++){

accu. [soFar = soFar && a[i] > 0;



i	soFar	a[i] > 0	accu.
0	false	3 > 0 True	false
1	false	4 > 0 True	false

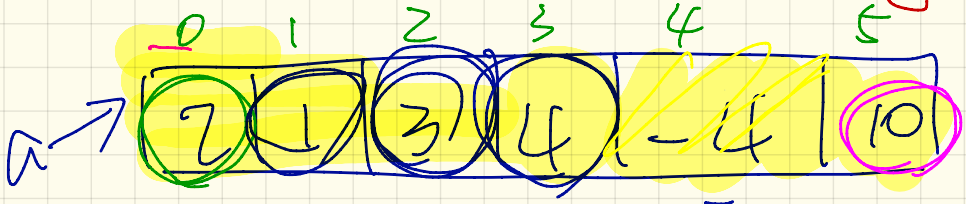
max = a[0]

0 X

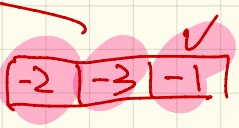
array contains all negatives

Exercise: find min.

```
1  int max = a[0];
2  for(int i = 0; i < a.length; i ++) {
3     if (a[i] > max) { max = a[i]; } }
4  System.out.println("Maximum is " + max);
```



Arrays.sort(a)



max: 2 → 3 → 4

i	a[i] > max	
0	2 > 2	false
1	1 > 2	false
2	3 > 2	true
3	4 > 3	true

non-decreasing : $!(\_ > \_)$

Sorting $\_ \leq \_$ $[!(\_ = \_) \;||\; \_ \not< \_]$

ascending order (increasing)

$1 < 3 \leq 5 < 6 < 7 < 8$ ✓

$1 \quad \boxed{3 \;\; 3} \quad \boxed{4 \;\; 4} \quad 5$ ✗

non-ascending order

$8 \geq 7 \geq 6 \geq 5 \geq 3 \geq 1$ ✓

$5 \geq 4 \geq 4 \geq 3 \geq 3 \geq 1$ ✓

descending order (decreasing)

$8 > 7 > 6 > 5 > 3 > 1$ ✓

$5 \not> 4 \not> 4 > 3 > 3 > 1$ ✗

non-descending order

✓ $1 \leq 3 \leq 5 \leq 6 \leq 7 \leq 8$

✓ $1 \leq 3 \leq 3 \leq 4 \leq 4 \leq 5$

Version 1: scan entire array

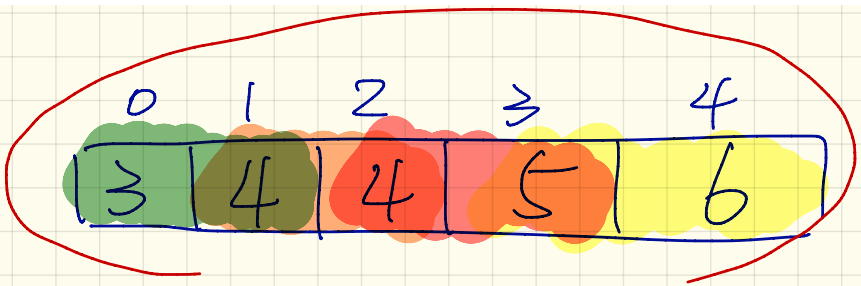
```
1  boolean isSorted = true;
2  for(int i = 0; i < a.length - 1; i ++) {
3    isSorted = isSorted && (a[i] <= a[i + 1]);
4  }
```

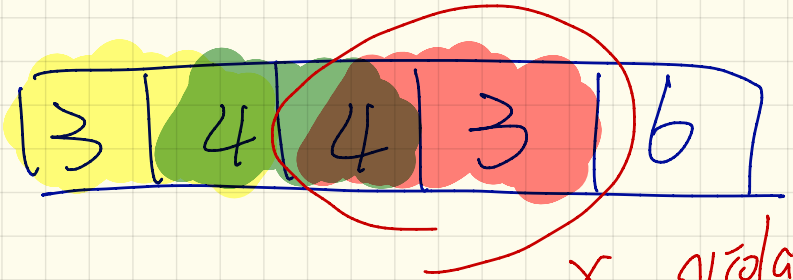
what if: `i < a.length`

i: 0 1 2 3 4

i from 0 to a.length - 2

i + 1





a[0] ≤ a[1]
a[1] ≤ a[2]
a[2] ≤ a[3]
a[3] ≤ a[4]
a[4] ≤ a[5]

x violation ∴ false

Version 2

As soon as wit. of violation is found, exit. a.length

```
1  boolean isSorted = true;
2  for(int i = 0; isSorted && i < a.length - 1; i ++) {
3    isSorted = a[i] <= a[i + 1];
4  }
```

false exits

if (a[i] > a[i+1]) isSorted = false

a.length is 5

i	
0	a[0] <= a[1]
1	a[1] <= a[2]
2	a[2] <= a[3]

0	1	2	3	4
3	4	4	3	6

false

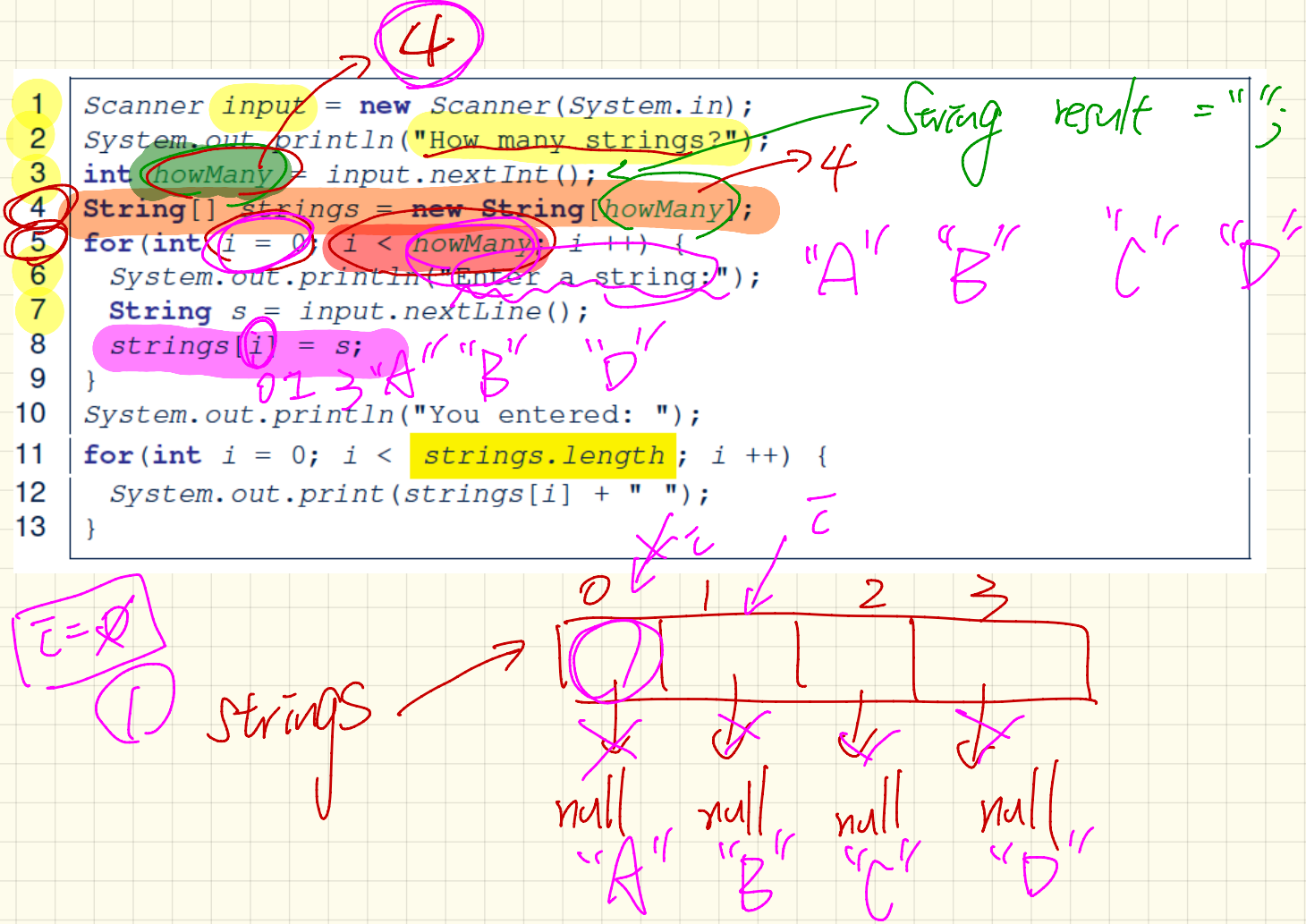
Version 1

```
boolean isSorted = true;
for (int i = 0; i < a.length - 1; i++){
    isSorted = isSorted && a[i] <= a[i+1];
```

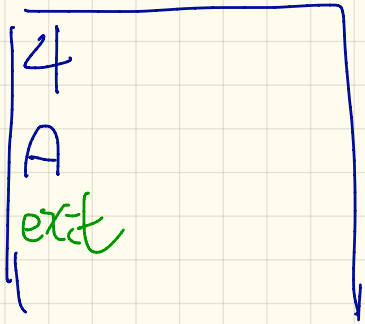
i	
0	a[0] <= a[1]
1	a[1] <= a[2]
2	a[2] <= a[3]
3	a[3] <= a[4]

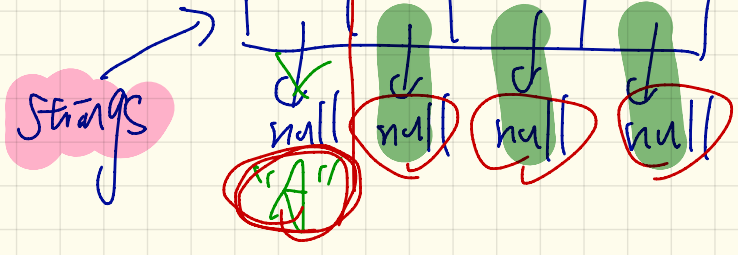
4 iterations

```
1  Scanner input = new Scanner(System.in);
2  System.out.println("How many strings?");
3  int howMany = input.nextInt();
4  String[] strings = new String[howMany];
5  for(int i = 0; i < howMany; i ++) {
6    System.out.println("Enter a string:");
7    String s = input.nextLine();
8    strings[i] = s;
9  }
10 System.out.println("You entered: ");
11 for(int i = 0; i < strings.length; i ++) {
12   System.out.print(strings[i] + " ");
13 }
```



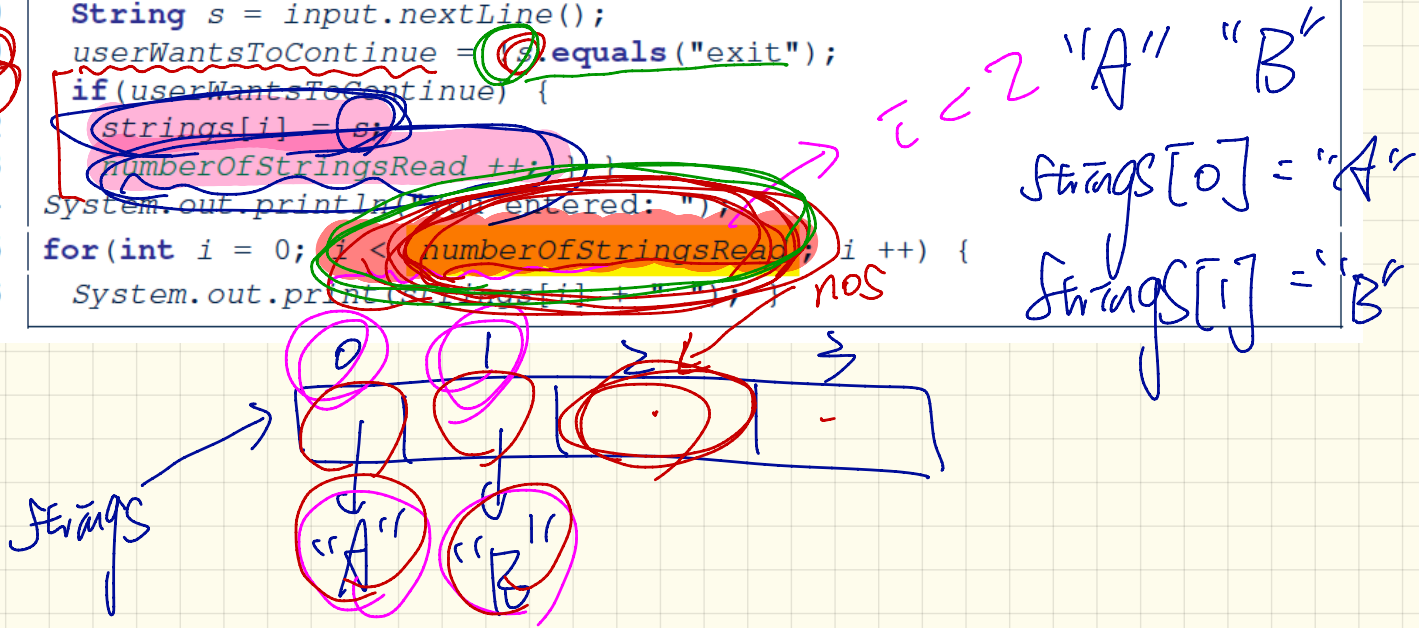
```
1  Scanner input = new Scanner(System.in);
2  System.out.println("How many strings?");
3  int howMany = input.nextInt();
4  boolean userWantsToContinue = true;
5  String[] strings = new String[howMany];
6  for(int i = 0; i < howMany && userWantsToContinue; i ++) {
7    System.out.println("Enter a string:");
8    String s = input.nextLine();
9    userWantsToContinue = !s.equals("exit");
10   if(userWantsToContinue) { strings[i] = s; }
11 }
12 System.out.println("You entered: ");
13 for(int i = 0; i < strings.length; i ++) {
14   System.out.print(strings[i] + " ");
15 }
```



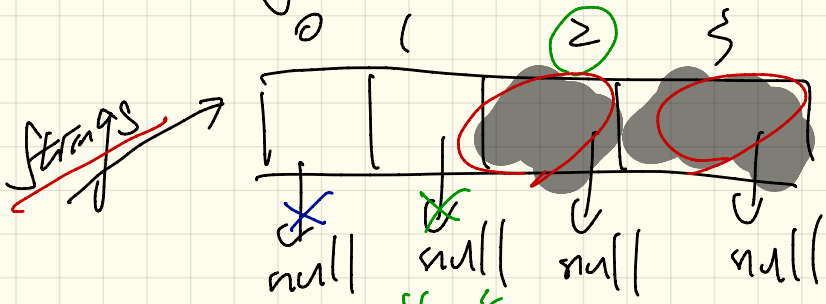


Version 3: Extended from V2.

```
1  Scanner input = new Scanner(System.in);
2  System.out.println("How many strings?");
3  int howMany = input.nextInt();
4  boolean userWantsToContinue = true;
5  String[] strings = new String[howMany];
6  int numberOfStringsRead = 0;
7  for(int i = 0; i < howMany && userWantsToContinue; i ++) {
8    System.out.println("Enter a string:");
9    String s = input.nextLine();
10   userWantsToContinue = !s.equals("exit");
11   if(userWantsToContinue) {
12     strings[i] = s;
13     numberOfStringsRead ++; } }
14 System.out.println("You entered: ");
15 for(int i = 0; i < numberOfStringsRead; i ++) {
16   System.out.print(strings[i] + " "); }
```



howMany 4



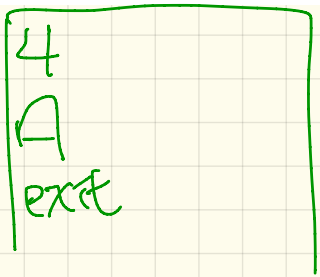
numberOfStrings

for(int i=0; i<nos; i++){
println(strings[i]);
}

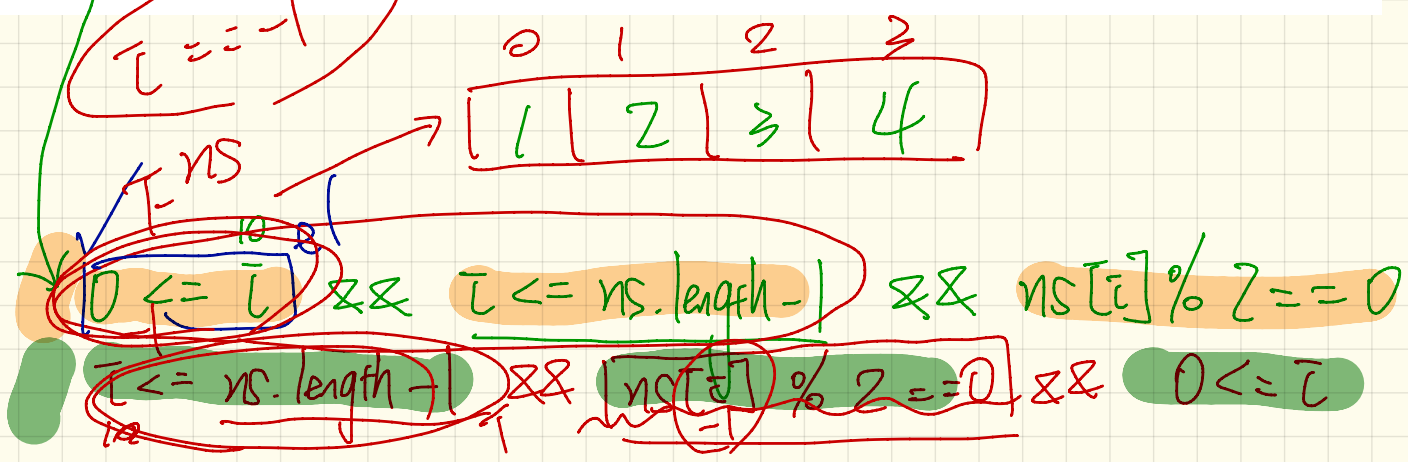
- I string read so far
- store next string at index 1

2 strings read
next string 2

```
1  Scanner input = new Scanner(System.in);
2  System.out.println("How many strings?");
3  int howMany = input.nextInt();
4  boolean userWantsToExit = false;
5  String[] strings = new String[howMany];
6  int numberOfStringsRead = 0;
7  for(int i = 0; i < howMany && !userWantsToExit; i ++) {
8    System.out.println("Enter a string:");
9    String s = input.nextLine();
10   userWantsToExit = s.equals("exit");
11   if(!userWantsToExit) {
12     strings[i] = s;
13     numberOfStringsRead ++; } }
14 System.out.println("You entered: ");
15 for(int i = 0; i < numberOfStringsRead; i ++) {
16   System.out.print(strings[i] + " "); }
```



```
1  Scanner input = new Scanner(System.in);
2  System.out.println("How many integers?");
3  int howMany = input.nextInt();
4  int[] ns = new int[howMany];
5  for(int i = 0; i < howMany; i ++) {
6    System.out.println("Enter an integer");
7    ns[i] = input.nextInt(); }
8  System.out.println("Enter an index:");
9  int i = input.nextInt();
10 if(ns[ i ] % 2 == 0) {
11   System.out.println("Element at index " + i + " is even."); }
12 else { /* Error ∵ ns[i] is odd */ }
```



Short Circuit