

Wednesday February 6

Lecture 10

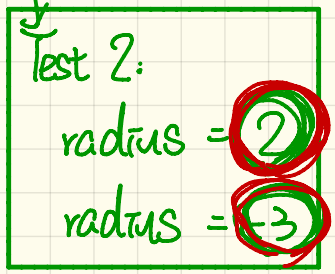
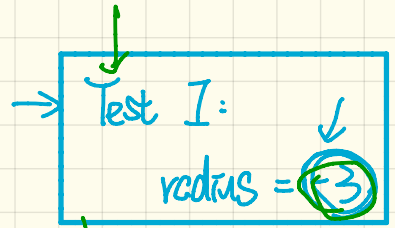
Compound Loop: Exercise (2.1)

```

1 System.out.println("Enter a radius value:");
2 double radius = input.nextDouble();
3 boolean isPositive = radius >= 0;
4 while (isPositive) {
5     → double area = radius * radius * 3.14;
6     → System.out.println("Area is " + area);
7     → System.out.println("Enter a radius value:");
8     → radius = input.nextDouble();
9     → isPositive = radius >= 0; }
10 System.out.println("Error: negative radius value.");
    
```

Annotations:

- Line 3: $\text{radius} \geq 0$ (True)
- Line 4: isPositive (True) → ②
- Line 5: $2 \times 2 \times 3.14$
- Line 9: $\text{isPositive} = \text{radius} \geq 0$ (True)

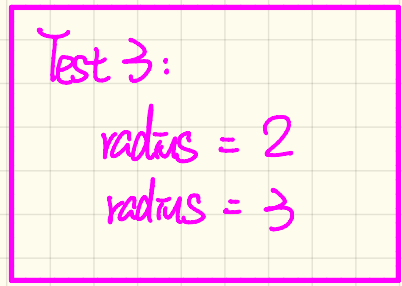


```

1 System.out.println("Enter a radius value:");
2 double radius = input.nextDouble();
3 boolean isNegative = radius < 0;
4 while (isNegative) {
5     → double area = radius * radius * 3.14;
6     → System.out.println("Area is " + area);
7     → System.out.println("Enter a radius value:");
8     → radius = input.nextDouble();
9     → isNegative = radius < 0; }
10 System.out.println("Error: negative radius value.");
    
```

Annotations:

- Line 3: isNeg (True)
- Line 4: isNegative (True) → ① isNeg (True)
- Line 5: $2 \times 2 \times 3.14$
- Line 9: $\text{isNegative} = \text{radius} < 0$ (True)



Compound Loop: Exercise (2.2)

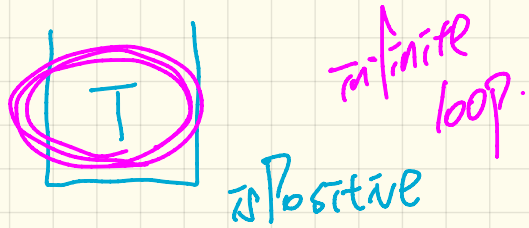
Q: What if we delete the update at Line 9?

Test 2:
radius = 2
radius = -3

```
1 System.out.println("Enter a radius value:");
2 double radius = input.nextDouble();
3 boolean isPositive = radius >= 0;
4 while (isPositive) {
5     double area = radius * radius * 3.14;
6     System.out.println("Area is " + area);
7     System.out.println("Enter a radius value:");
8     radius = input.nextDouble();
9     radius = input.nextDouble();
10 System.out.println("Error: negative radius value.");
```

$-3 * -3 * 3.14$
 $2 * 2 * 3.14$

Console:



for-loop \longleftrightarrow while-loop

- To convert a `while` loop to a `for` loop, leave the initialization and update parts of the `for` loop empty.

```
while (B) {  
    /* Actions */  
}
```

is equivalent to:

```
for( ; B; ) {  
    /* Actions */  
}
```

- To convert a `for` loop to a `while` loop, move the initialization part immediately before the `while` loop and place the update part at the end of the `while` loop body.

```
for(int i = 0; B; i++) {  
    /* Actions */  
}
```

is equivalent to:

```
int i = 0;  
while (B) {  
    /* Actions */  
    i++;  
}
```

Stay Condition vs. Exit Condition

P & Q

$$\!(p \&\& q) == !p \parallel !q$$

- When does the loop exit (i.e., stop repeating Action 1)?

```
→ while (p && q) { /* Action 1 */ }
```

→ $\!(p \&\& q)$: exit condition
exit condition = $\!(\text{stay condition})$

- When does the loop exit (i.e., stop repeating Action 2)?

```
→ while (p || q) { /* Action 2 */ }
```

→ $\!(p \parallel q)$: exit condition
 $\!(p \parallel q) == !p \&\& !q$

Stay Condition vs. Exit Condition: Exercise

Consider the following loop:

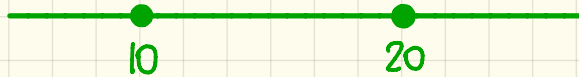
```
int x = input.nextInt();  
while(10 <= x || x <= 20) {  
    → /* body of while loop */  
}
```

- It compiles, but has a logical error. Why?

→ Stay Condition

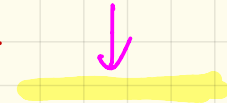
while (T) X

always repeat action.



(T)

→ Exit Condition



$10 > x$ && $x > 20$

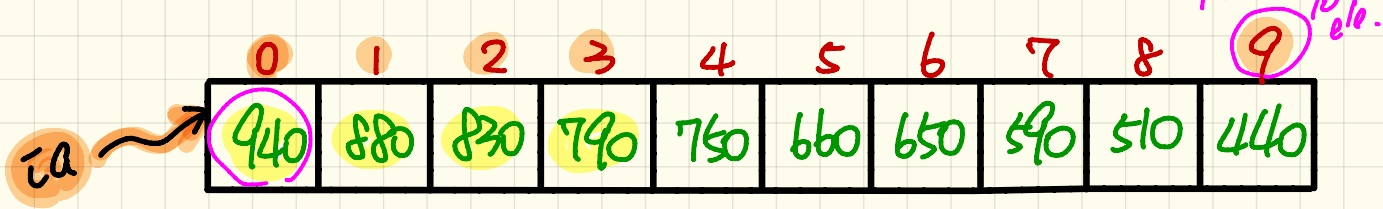
(F)

!(10 <= x || x <= 20)

→ (!)(10 <= x) && !(x <= 20)

Array of Integers (1)

No pattern on stored values



Declaration and Initialization: Approach 1 (Initializer)

```
int [] ia = { 940, 880, 830, 790, 750, 660, 650, 590, 510, 440 };
```

Declaration and Initialization: Approach 2 (Assignments)

```
int [] ia = new int [ 10 ]; ← an array of int of size 10  
ia[0] = 940; ia[1] = 880; ... ;  
ia[9] = 440;
```

Diagram illustrating the array structure for Approach 2. The array is represented as a horizontal row of 10 cells, indexed from 0 to 9. The first cell (index 0) contains the value 940 and is circled in pink. The last cell (index 9) is also circled in pink. A bracket above the last cell is labeled "ia.length - 1" and "size 10".