Name: _____ Session: AM / PM

Circle one

Chapter 4 - Day 1

Reading Loops: ++, while, and do-while

INCREMENT AND DECREMENT

++ and --: ++ adds 1 to a variable and -- subtracts 1. Prefix mode puts the operator before the variable (++x); postfix mode puts it after (x++). On a line by itself, either mode _____.

Term	Definition / memory cue
increment operator	
decrement operator	
prefix mode	
postfix mode	

E1 Standalone ++ and -- `MissionCountersStep1.java`

What changed: Three counters start at 5. Each one changes by 1 in a statement by itself, then prints.

```
4         int probesA = 5;
5         probesA++;
6         System.out.println("probesA = " + probesA);
7
8         int probesB = 5;
9         ++probesB;
10        System.out.println("probesB = " + probesB);
11
12        int cells = 5;
13        cells--;
14        System.out.println("cells = " + cells);
```

1. Line 5 uses _____ mode
2. Line 9 uses _____ mode

Predict each line E1 prints.

Line	Output	Ran it?
`6`		☐
`10`		☐
`14`		☐

When the mode matters: Inside a larger expression, such as an assignment, the mode decides the order. Postfix (x++) uses the current value, then adds 1. Prefix (++x) adds 1, then _____ .

E2 Inside an assignment

`MissionCountersStep2.java`

What changed: Two more counters start at 3. Each one changes inside an assignment, then the stored value and the counter both print.

```
16          // Inside an assignment: postfix uses, then changes
17          int launchesA = 3;
18          int loggedA = launchesA++;
19          System.out.println("loggedA = " + loggedA);
20          System.out.println("launchesA = " + launchesA);
21
22          // Prefix changes, then uses
23          int launchesB = 3;
24          int loggedB = ++launchesB;
25          System.out.println("loggedB = " + loggedB);
26          System.out.println("launchesB = " + launchesB);
```

1. Line 18 stores launchesA's value _____ adding 1
2. Line 24 stores launchesB's value _____ adding 1

Predict each value E2 prints.

Variable	Value	Ran it?
`loggedA`		☐
`launchesA`		☐
`loggedB`		☐
`launchesB`		☐

THE WHILE LOOP

How a while loop works: A while loop checks its condition first. When the condition is true, the body runs, then the condition is checked again. When the condition is _____, the loop ends and the program continues after the closing brace. Because the check comes first, the body might run _____ times.

Term	Definition / memory cue
loop	
iteration	
loop body	
loop control variable	
pretest loop	

Loop reading checklist: Ask four questions about every loop. (1) Which variable controls it, and where does it start? (2) What is the condition? (3) Which lines are in the body? (4) Which line changes the control variable, and will the condition ever become false?

E4 Anatomy of a while loop

`SignalPing.java`

What changed: A loop prints a ping while ping is 3 or less, then prints a done message. Annotate it with the key before it runs.

```
3         // A while loop: check the condition, run the body, repeat
4         int ping = 1;
5         while (ping <= 3) {
6             System.out.println("Ping " + ping);
7             ping++;
8         }
9         System.out.println("Signal check done.");
```

1. Loop control variable:

2. Line _____ changes the control variable

3. The body runs _____ times

Trace tables: A trace table records each variable's value every time the condition is checked. Write one row per check, including the last check, the one that is _____.

E5 Two variables, one loop

AltitudeClimb.java

What changed: Each iteration adds stage * 100 to altitude, prints both, then moves to the next stage.

```
3         // Two variables change on every iteration
4         int stage = 1;
5         int altitude = 0;
6         while (stage < 4) {
7             altitude += stage * 100;
8             System.out.printf("Stage %d: %d km%n", stage, altitude);
9             stage++;
10        }
11        System.out.println("Final stage value: " + stage);
```

1. Changing < to <= on line 6 runs _____ iteration

Trace E5 with your partner: one row per condition check. Row 1 is done together.

stage	altitude	stage < 4	Prints next
1	0		

When loops go wrong: An infinite loop never ends because its condition never becomes _____. A semicolon right after the `while` header is an empty statement, and it becomes the whole body. Without braces, the body is only the next _____, no matter how the code is indented.

Finger vote. Each version is E4 with one change. Write your vote (1-4), then what actually happened.

Version	The change	Vote	What happened
E6a	`int ping = 4;`		
E6b	`while (ping <= 3);`		
E6c	Braces removed		

E6c: What is really in the body?

Bracket what the loop body really holds.

```
1 int ping = 1;
2 while (ping <= 3)
3     System.out.println("Ping " + ping);
4     ping++;
5 System.out.println("Signal check done.");
```

1. The loop body is only line

2. Line 4 is indented, but it is _____ the loop

3. ping never changes, so the condition stays

INPUT VALIDATION

Input validation: Input validation checks what the user typed and asks again until it is valid. A priming read gets the first value _____ the loop, so the condition has something to check. A second read _____ the loop gets each new try.

Term	Definition / memory cue
input validation	
priming read	

E7-1 The priming read

`CrewSizeStep1.java`

What changed: Reads the crew size once and echoes it. There is no check yet.

```
7         // Priming read: get a value before any loop checks it
8         System.out.print("Enter the crew size (1-6): ");
9         int crew = keyboard.nextInt();
10
11        System.out.println("Crew size set to " + crew + ".");
```

1. The priming read is line

2. Type 9, and this step prints

E7-2 The validation loop

CrewSizeStep2.java

What changed: Adds a while loop after the priming read. It repeats while the crew size is outside 1 to 6.

```
7         // Priming read: get a value before any loop checks it
8         System.out.print("Enter the crew size (1-6): ");
9         int crew = keyboard.nextInt();
10
11        // Validation loop: repeat while the input is invalid
12        while (crew < 1 || crew > 6) {
13            System.out.println("Invalid: the capsule holds 1 to 6.");
14            System.out.print("Enter the crew size (1-6): ");
15            crew = keyboard.nextInt();
16        }
17
18        System.out.println("Crew size set to " + crew + ".");
```

1. The condition on line 12 is true while the input is

2. Line _____ reads each new try

3. If the first input is 4, the body runs _____ times

Scripted input: 0, then 9, then 4. Predict what prints after each one.

You type	What prints next	Ran it?
0		☐
9		☐
4		☐

Finger vote. Each version is E7-2 with one change. Write your vote (1-4), then what actually happened.

Version	The change	Vote	What happened
E8a	No priming read: crew is declared with no value		
E8b	No read inside the loop (line 15 removed)		

THE DO-WHILE LOOP

How a do-while loop works: A do-while loop runs its body first, then checks its condition, so the body always runs at least _____. The condition sits after the closing brace and ends with a _____.

Term	Definition / memory cue
do-while loop	
posttest loop	
user-controlled loop	

E9 A user-controlled loop

CargoLog.java

What changed: Logs one crate per iteration, then asks whether to log another. The first character of the answer decides whether the loop repeats.

```
6         int crate = 0;
7         String input;
8         char again;
9
10        // do-while: the body runs first, then the condition is checked
11        do {
12            crate++;
13            System.out.print("Crate " + crate + " weight (kg): ");
14            double weight = keyboard.nextDouble();
15            System.out.printf("Crate %d: %.1f kg%n", crate, weight);
16
17            keyboard.nextLine(); // throw away the leftover newline
18            System.out.print("Log another crate? (Y/N): ");
19            input = keyboard.nextLine();
20            again = input.charAt(0);
21        } while (again == 'Y');
22
23        System.out.println("Crates logged: " + crate);
```

1. Line _____ throws away the leftover newline
2. The condition is checked on line
3. Why is again declared on line 8, before do?

Trace E9 with the scripted input 120, Y, 85.5, N: one row per iteration.

Iteration	crate	weight	again	again == 'Y'
1				
2				

Predict the last line E9 prints.

Input typed	Last line	Ran it?
`120, Y, 85.5, N`		☐
`120, y`		☐

WHILE VS. DO-WHILE

Compare the two loops.

Question	while	do-while
When is the condition checked?		
Fewest times the body runs		
Pretest or posttest?		

E10 Same body, two loops

`LoopShowdown.java`

What changed: Both loops start at 10 and check < 3, so both conditions are false from the start.

```
3         // Same body, same false condition, two different loops
4         int burns = 10;
5         while (burns < 3) {
6             System.out.println("while loop: burn " + burns);
7             burns++;
8         }
9
10        int pulses = 10;
11        do {
12            System.out.println("do-while loop: pulse " + pulses);
13            pulses++;
14        } while (pulses < 3);
15
16        System.out.println("burns = " + burns + ", pulses = " + pulses);
```

1. Which loop's body runs, and why?

Predict every line E10 prints. Write (nothing) if a loop prints nothing.

Source	Output	Ran it?
`while loop`		☐
`do-while loop`		☐
`line 16`		☐