

The Loop Toolbox

Implementing selection and iteration algorithms



One receipt at a time

PAUSE AND TRY

You see these receipts one at a time. What do you remember between them?

\$4

\$7

\$2

\$5

1. What is the total?
2. Which receipt was the cheapest?
3. How many numbers did you have to keep in your head?

By the end of class you can...

- 1 Add up a sum with an accumulator, and compute an average that keeps its decimals.
- 2 Count how many values meet a condition with an if inside a loop.
- 3 Test whether one integer is evenly divisible by another with %.
- 4 Pull an integer apart into its digits with % 10 and / 10.
- 5 Find a minimum or maximum, starting from a value the data will replace.

A large, white, rounded square containing the number '1' in a bold, teal, sans-serif font.

1

PART 1

Sums and averages

The running total

ADD THE ODD NUMBERS 1 TO 9

```
int total = 0;
for (int odd = 1; odd <= 9; odd += 2) {
    total += odd;
}
System.out.println(total);
```

OUTPUT

25

Before the loop: start the accumulator at 0, once.

Inside the loop: add one value with +=.

After the loop: use the finished total.

An **accumulator** collects a result a little at a time as the loop runs.

Average of five quiz scores

QUIZAVERAGE

```
Scanner console = new Scanner(System.in);
int sum = 0;
for (int quiz = 1; quiz <= 5; quiz++) {
    sum += console.nextInt();
}
System.out.println(sum / 5);
System.out.println((double) (sum / 5));
System.out.println((double) sum / 5);
```

INPUT

88 92 75 90 81

OUTPUT

85

85.0

85.2

The sum is 426.

Average = $\text{sum} \div \text{number of values}$. Same sum, three different answers. Which is right?

Cast before you divide



int / int drops the decimal part. Cast **one value to double before** the division happens.

Expression	What Java does	Result
sum / 5	int ÷ int: decimal part dropped	85
(double) (sum / 5)	Divides first (85), then casts	85.0
(double) sum / 5	Casts first: 426.0 ÷ 5	85.2

No values? If the count can be 0, check count > 0 before you divide.

A large, white, rounded square containing the number '2' in a bold, teal, sans-serif font.

2

PART 2

Counting and divisibility

Count when the test says yes

MULTIPLES OF 4 UP TO 30

```
int count = 0;
for (int n = 1; n <= 30; n++) {
    if (n % 4 == 0) {
        count++;
    }
}
System.out.println(count);
```

OUTPUT

7



Frequency: start a counter at 0 before the loop; add 1 only when an if finds the condition true.



Divisibility: $a \% b == 0$ is true exactly when a is evenly divisible by b. Even: $n \% 2 == 0$.

WORKED EXAMPLE

Every divisor of 18

```
int number = 18;
int divisors = 0;
for (int d = 1; d <= number; d++) {
    if (number % d == 0) {
        System.out.print(d + " ");
        divisors++;
    }
}
System.out.println();
System.out.println("Divisors: " + divisors);
```

d	18 % d	Divisor?
1, 2, 3	0, 0, 0	yes
4, 5	2, 3	no
6	0	yes
7, 8	4, 2	no
9	0	yes
10 to 17	not 0	no
18	0	yes

OUTPUT

```
1 2 3 6 9 18
Divisors: 6
```

Which number goes first?



Read `a % b == 0` as "a divided by b leaves nothing." The thing being shared goes **on the left**.

IS 96 DIVISIBLE BY 8?

```
System.out.println(8 % 96);  
System.out.println(96 % 8);
```

OUTPUT

```
8  
0
```

8 % 96 is 8: 96 goes into 8 zero times, so all 8 is left over. The test fails.

96 % 8 is 0: $8 \times 12 = 96$. Divisible.

Not / ! $96 / 8$ is 12, the quotient. Only % gives a remainder.

Count with two tests

PAUSE AND TRY

What does this print? List the numbers it counts.

```
int count = 0;
for (int n = 1; n <= 50; n++) {
    if (n % 5 == 0 && n % 10 != 0) {
        count++;
    }
}
System.out.println(count);
```

Hint: divisible by 5, but not by 10.

A white rounded square with a teal number '3' in the center.

3

PART 3

Taking apart the digits

% 10 takes the last digit; / 10 drops it

Expression (number is 731)	Meaning	Value
<code>number % 10</code>	the last digit (ones)	1
<code>number / 10</code>	drop the last digit	73
<code>number / 10 % 10</code>	the tens digit	3



To visit **every** digit, loop **while (n > 0)**: take the last digit with `% 10`, then drop it with `n /= 10`. The digits come out right to left.

Print the digits of 2581, backward

```
int n = 2581;
int digits = 0;
while (n > 0) {
    System.out.print(n % 10);
    digits++;
    n /= 10;
}
System.out.println();
System.out.println(digits + " digits");
```

OUTPUT

```
1852
4 digits
```

n at start	prints n % 10	digits	n after /= 10
2581	1	1	258
258	8	2	25
25	5	3	2
2	2	4	0

n is 0, so $0 > 0$ is false and the loop ends.

How many digits does 0 have?



A **while** (**n > 0**) loop never runs when n starts at 0. **Test the edge cases.**

```
int n = 0;
int digits = 0;
while (n > 0) {
    digits++;
    n /= 10;
}
System.out.println(digits);
```

OUTPUT

0

Your turn: add one if statement before the loop so the program prints 1 for 0 and still works for every positive number.

A large, bold, teal-colored number '4' centered within a white rounded square with a slight drop shadow.

4

PART 4

The best so far

Keep the best so far

```
Scanner console = new Scanner(System.in);  
int highest = console.nextInt();  
for (int player = 2; player <= 5; player++) {  
    int score = console.nextInt();  
    if (score > highest) {  
        highest = score;  
    }  
}  
System.out.println("Highest: " + highest);
```

INPUT

54 81 77 96 62

OUTPUT

Highest: 96

Start: the first value, or a value every real value beats
(Integer.MIN_VALUE for a max,
Integer.MAX_VALUE for a min).

Update: max uses >, min uses <.



Starting the best so far at 0



A starting value the data can't beat **never gets replaced**. Start with real data, or a value every real value beats.

```
Scanner console = new Scanner(System.in);
int cheapest = 0;
for (int item = 1; item <= 3; item++) {
    int price = console.nextInt();
    if (price < cheapest) {
        cheapest = price;
    }
}
System.out.println("Cheapest: " + cheapest);
```

INPUT

12 5 9

OUTPUT

Cheapest: 0

No item cost 0. The cheapest was 5.

Two jobs, one loop

PAUSE AND TRY

What does this print? Trace *v*, *most*, and *total*.

```
int most = Integer.MIN_VALUE;
int total = 0;
for (int k = 1; k <= 4; k++) {
    int v = k * 5 % 8;
    total += v;
    if (v > most) {
        most = v;
    }
}
System.out.println(most + " " + total);
```

k	v	most	total
1			
2			
3			
4			

Words to know

Standard algorithm

A reusable step-by-step solution to a common problem.

Accumulator

A variable that builds a result a little at a time, like a running total.

Counter

An accumulator that adds 1 each time something happens.

Average

The sum divided by the number of values.

Evenly divisible

a is evenly divisible by b when $a \% b == 0$.

Minimum / maximum

The smallest / largest value, found with a best-so-far variable.

Pick the right tool



A game wants the total points, the number of perfect rounds, and the best round. Tell your partner how each variable starts and changes.

SENTENCE STARTER

The _____ starts at _____ before the loop, and on each pass it _____.

Next: The Loop Toolbox puzzle worksheet.