

Transcript for Digits:

Dijiste:

how would i separate the digits of a two digit number in java?

ChatGPT dijo:

You can separate the digits of a two-digit number in Java using integer division (/) and the modulus operator (%).

Here's a simple example:

```
public class SplitDigits {
    public static void main(String[] args) {
        int number = 47; // example two-digit number

        int tens = number / 10;    // get the tens place
        int ones = number % 10;    // get the ones place

        System.out.println("Tens digit: " + tens);
        System.out.println("Ones digit: " + ones);
    }
}
```

Output for 47:

```
Tens digit: 4
Ones digit: 7
```

✅ Works for any two-digit number (10–99).

Do you want me to also show you how to do it if the number comes in as a **String** instead of an **int**?

Dijiste:

```
in this project what is the warning and how do i fix it? or do i need to fix it? package skillBuilders;
import java.util.Scanner; class Digits {      public static void main(String[] args) {
//Get user input          Scanner input = new Scanner(System.in);
System.out.print("Enter a two digit number: ");          int userNum = input.nextInt();
                        //Separate the number into 2 int tens = userNum / 10; int ones = userNum
% 10;          System.out.println("Ones-place digit: " + ones); System.out.println("Tens-place
digit: " + tens);          } }
```

ChatGPT dijo:

In your project, the only likely warning is about the `Scanner` not being closed:

Resource leak: 'input' is never closed

This happens because `Scanner` implements `AutoCloseable`, and IDEs like Eclipse will warn you when you don't call `input.close()`.

Do you need to fix it?

- **For small programs like this (short-lived, ends immediately):** It's not a real problem. The JVM will clean everything up once the program exits, so you can safely ignore the warning.
- **For larger projects / good practice:** Yes, you should close it.

How to fix

Option 1: Manually close the scanner at the end:


```
```java
input.close();
```

Option 2 (preferred in modern Java): Use *try-with-resources*, which auto-closes:

```
try (Scanner input = new Scanner(System.in)) {
 System.out.print("Enter a two digit number: ");
 int userNum = input.nextInt();

 int tens = userNum / 10;
 int ones = userNum % 10;

 System.out.println("Ones-place digit: " + ones);
 System.out.println("Tens-place digit: " + tens);
}
```

 **Note:** Closing a `Scanner` on `System.in` will also close `System.in` itself. That means if you later want to use `System.in` again in the same program, it won't work. For simple one-shot programs like this, it's fine.

---

Do you want me to rewrite your version with the `try-with-resources` pattern so it's 100% clean and warning-free?

**Dijiste:**

what is a resource leak

**ChatGPT dijo:**

A **resource leak** happens when your program takes control of a system resource (like memory, files, network connections, or input/output streams) but never gives it back to the operating system.

In Java, common resources include:

- Files (`FileInputStream`, `FileWriter`, etc.)
- Database connections
- Network sockets
- Input/output streams (like `Scanner`, `BufferedReader`)

### Why does this matter?

- If your program runs for a short time and then exits (like your `Digits` example), the OS will clean everything up when the JVM shuts down. The leak isn't dangerous here.
- If your program runs for a long time (like a server), unreleased resources can accumulate, eventually causing the program or system to run out of memory, file handles, or connections. That's when resource leaks cause real problems.

### Example

```
```java
// Potential resource leak
Scanner input = new Scanner(System.in);
int n = input.nextInt();
// program ends without input.close();
```

⚠ The IDE warns you because you didn't call `input.close()`. That means the resource (System.in stream) stays open until the JVM exits.

Fix

Use `close()` or `try-with-resources`:

```
try (Scanner input = new Scanner(System.in)) {  
    int n = input.nextInt();  
}
```

This guarantees that `input` will be closed automatically, even if an error happens.

So in your program:

- **Warning only:** You don't *need* to fix it for a small console app.
- **Good habit:** Fix it in real-world or long-running apps to avoid actual leaks.

Would you like me to show you a simple analogy (like a "borrowed library book" comparison) to make resource leaks easier to picture?

Transcript for Sleep:

Dijiste:

If i was born on February 4th, 2009, and the current date is Sept. 15th, 2025, assuming every year has 365 days and every month has 30 days, how many days have I lived for?

ChatGPT dijo:

Pensado durante 8s

Quick answer (with the assumptions you gave):

- ****Days lived = 6,061 days**** (counting from Feb 4, 2009 up to Sep 15, 2025, where a year = 365 days and a month = 30 days).
 - If you want to include the birth day itself as a counted day, add 1
- ****6,062 days****.

Work shown (step-by-step using your simplified calendar):

1. Whole years from Feb 4, 2009 → Feb 4, 2025 = 16 years.
16 years × 365 days/year = 5,840 days.

2. Remaining time from Feb 4, 2025 → Sep 15, 2025:
- Feb 4 → Sep 4 = 7 months (Mar, Apr, May, Jun, Jul, Aug, Sep)
7 months × 30 days/month = 210 days
 - Sep 4 → Sep 15 = 11 days
 - Extra = 210 + 11 = 221 days

3. Total = 5,840 + 221 = 6,061 days.

Notes:

- I used your exact rules (every year = 365 days, every month = 30 days).
- If you prefer to count the birth day itself as a lived day, add 1 → **6,062**.

Dijiste:

Assuming I sleep for 8 hours each day, how many hours have I slept total

ChatGPT dijo:

You've lived 6,061 days (by the simplified calendar rules).

Sleep per day = 8 hours.

Total sleep = 6,061 × 8 hours
= 48,488 hours.

If you also counted the birth day (making it 6,062 days), then:
6,062 × 8 = 48,496 hours.

Transcript for Election:

Dijiste:

```
Here is some code i made: package mastery; import java.text.NumberFormat; import
java.util.Scanner; public class Election {      public static void main(String[] args) {
//Prompt for election results      //BC      Scanner input = new Scanner(System.in);
      System.out.println("Elction Results for BC: ");      System.out.print("John: ");
      int johnVotesBC = input.nextInt();      System.out.print("Carrie: ");      int
carrieVotesBC = input.nextInt();      System.out.println("");
//Alberta      System.out.println("Elction Results for Alberta: ");
      System.out.print("John: ");      int johnVotesAB = input.nextInt();
```

```

System.out.print("Carrie: ");          int carrieVotesAB = input.nextInt();
System.out.println("");                //Saskatchewan
System.out.println("Election Results for Saskatchewan: ");      System.out.print("John: ");
    int johnVotesSK = input.nextInt();      System.out.print("Carrie: ");      int
carrieVotesSK = input.nextInt();      System.out.println("");
    //Calculations      int johnTotal = johnVotesBC + johnVotesAB +
johnVotesSK;      int carrieTotal = carrieVotesBC + carrieVotesAB + carrieVotesSK;
int totalVotes = johnTotal + carrieTotal;      //Display final results and
calculate percentages      NumberFormat percent =
NumberFormat.getPercentInstance();      System.out.format("%-10s %8s %8s",
"Candidate", "Votes", "Percentage");      System.out.format("%-10s %8s %8s", "John", +
johnTotal, percent.format(johnTotal));      System.out.format("%-10s %8s %8s",
"Carrie", + carrieTotal, percent.format(carrieTotal));      System.out.format("%-10s %8s",
"TOTAL VOTES:" + totalVotes);    } } And I am getting this error, how to fix: Exception in
thread "main" java.util.MissingFormatArgumentException: Format specifier '%8s' at
java.base/java.util.Formatter.format(Formatter.java:2790) at
java.base/java.io.PrintStream.implFormat(PrintStream.java:1367) at
java.base/java.io.PrintStream.format(PrintStream.java:1346) at
mastery.Election.main(Election.java:45)

```

ChatGPT dijo:

The error happens because your `System.out.format()` calls have **more format specifiers (`%s`) than arguments** provided.
For example:

```

```java
System.out.format("%-10s %8s %8s", "Candidate", "Votes",
"Percentage");

```

This is fine (3 specifiers, 3 arguments).

But here:

```

System.out.format("%-10s %8s %8s", "John", + johnTotal,
percent.format(johnTotal));

```

You actually gave **3 arguments**, but notice:

- `+ johnTotal` isn't wrong, but `percent.format(johnTotal)` is formatting John's raw number instead of dividing by `totalVotes`.

Also, at the end:

```
System.out.format("%-10s %8s", "TOTAL VOTES:" + totalVotes);
```

This has **2 format specifiers** (**%-10s %8s**) but only **1 argument** provided. That's exactly why you're seeing `MissingFormatArgumentException`.

---

### ✓ Fix

- Make sure the number of % specifiers matches the number of arguments.
- Calculate percentages properly by dividing by `totalVotes`.

Here's a corrected version:

```
package mastery;
import java.text.NumberFormat;
import java.util.Scanner;

public class Election {
 public static void main(String[] args) {
 Scanner input = new Scanner(System.in);

 // BC
 System.out.println("Election Results for BC: ");
 System.out.print("John: ");
 int johnVotesBC = input.nextInt();
 System.out.print("Carrie: ");
 int carrieVotesBC = input.nextInt();
 System.out.println("");

 // Alberta
 System.out.println("Election Results for Alberta: ");
 System.out.print("John: ");
 int johnVotesAB = input.nextInt();
 System.out.print("Carrie: ");
 int carrieVotesAB = input.nextInt();
 System.out.println("");
 }
}
```

```

 // Saskatchewan
 System.out.println("Election Results for Saskatchewan: ");
 System.out.print("John: ");
 int johnVotesSK = input.nextInt();
 System.out.print("Carrie: ");
 int carrieVotesSK = input.nextInt();
 System.out.println("");

 // Calculations
 int johnTotal = johnVotesBC + johnVotesAB + johnVotesSK;
 int carrieTotal = carrieVotesBC + carrieVotesAB +
carrieVotesSK;
 int totalVotes = johnTotal + carrieTotal;

 // Display final results and calculate percentages
 NumberFormat percent = NumberFormat.getPercentInstance();

 System.out.format("%-10s %8s %12s%n", "Candidate", "Votes",
"Percentage");
 System.out.format("%-10s %8d %12s%n", "John", johnTotal,
percent.format((double) johnTotal / totalVotes));
 System.out.format("%-10s %8d %12s%n", "Carrie", carrieTotal,
percent.format((double) carrieTotal / totalVotes));
 System.out.format("%-10s %8d%n", "TOTAL VOTES:", totalVotes);

 input.close();
 }
}

```

---

### Changes made:

1. Used `%d` for integers instead of `%s`.
2. Divided vote counts by `totalVotes` for percentages.
3. Added `"%n"` to move to new lines.



4. Fixed the last line so it has **two arguments** for the **two format specifiers**.

---

Would you like me to also show you how to make the output appear in a **nice table format** (aligned rows/columns like election results)?

**Dijiste:**

why do you have to turn it into a double before making it a percentage

**ChatGPT dijo:**

Great question 👍

You have to cast to ``double`` because of how **integer division** works in Java.

---

### Integer division problem

If you do this:

```
```java
int johnTotal = 40;
int totalVotes = 100;
double percentJohn = johnTotal / totalVotes;
```

The division is done between two `ints` → so Java performs **integer division**, which drops the decimal part:

`40 / 100 = 0` (as an int)

Then it's assigned to `double`:

`percentJohn = 0.0`

That's why without the cast, you'd always get `0%` unless John got **all** the votes.

Solution: force floating-point division

By casting one operand to `double`, Java switches to floating-point division:

```
(double) johnTotal / totalVotes
```

Now:

```
40 / 100 = 0.4 (as a double)
```

Then `NumberFormat.getPercentInstance()` will format that properly as:

```
40%
```

✓ In short:

- **Without** `(double)` → integer division → `0%` (wrong).
 - **With** `(double)` → floating-point division → `40%` (correct).
-

Do you want me to show you what the output would look like **with and without the cast** so you can see the difference in the table?``

Dijiste:

why does the number of total votes not line up with the other votes in the table

ChatGPT dijo:

Ah, that's because of the **format specifiers** in your ``System.out.format()`` calls.

Let's look at the rows:

```
``java
System.out.format("%-10s %8d %12s%n", "John", johnTotal,
percent.format((double) johnTotal / totalVotes));
System.out.format("%-10s %8d %12s%n", "Carrie", carrieTotal,
percent.format((double) carrieTotal / totalVotes));
```

```
System.out.format("%-10s %8d%n", "TOTAL VOTES:", totalVotes);
```

Why the misalignment happens

- The **candidate rows** have **three columns**:
`%-10s` (name), `%8d` (votes), `%12s` (percentage).
 - The **TOTAL row** only has **two columns**:
`%-10s` (label), `%8d` (votes).
→ So there's no third column for the percentage, which makes it *look* shifted.
-

How to fix alignment

If you want the table to stay perfectly lined up, you can add an **empty percentage column** for the TOTAL row:

```
System.out.format("%-10s %8d %12s%n", "TOTAL VOTES:", totalVotes, "");
```

Now all three rows use the same format string: `%-10s %8d %12s`, so everything stays aligned in neat columns.

Example output

Candidate	Votes	Percentage
John	40	40%
Carrie	60	60%
TOTAL VOTES:	100	

Do you want me to show you how to make it look even more like a proper table, with lines or separators (like `|` and `----`), so it's easier to read?``

Dijiste:

```
//Display final results and calculate percentages
percent = NumberFormat.getPercentInstance();
System.out.format("%-10s %8s
```

```
%10s %n", "Candidate", "Votes", "Percentage");          System.out.format("%-10s %8s
%10s %n", "John", johnTotal, percent.format((double)johnTotal / totalVotes));
System.out.format("%-10s %8s %10s %n", "Carrie", carrieTotal,
percent.format((double)carrieTotal / totalVotes));          System.out.format("%-10s %8s
%10s %n", "TOTAL VOTES: ", totalVotes, ""); This is the code, the final one still is not aligned
```

ChatGPT dijo:

Got it 👍 - I see what's happening.

Your format string is:

```
```java
"%-10s %8s %10s %n"
```

That means:

- `%-10s` → left-align a string in a 10-character field
- `%8s` → right-align a string in an 8-character field
- `%10s` → right-align a string in a 10-character field

This works fine for "John" and "Carrie", but for the last row:

```
System.out.format("%-10s %8s %10s %n", "TOTAL VOTES: ", totalVotes,
"");
```

Problem:

- `"TOTAL VOTES:"` is **longer than 10 characters**.  
→ It spills past its column and pushes the numbers over, breaking the alignment.

---

## ✅ Fix

Give the first column more width so `"TOTAL VOTES:"` fits cleanly. For example:

```
System.out.format("%-15s %8s %10s %n", "Candidate", "Votes",
"Percentage");
```

```
System.out.format("%-15s %8d %10s %n", "John", johnTotal,
percent.format((double)johnTotal / totalVotes));
System.out.format("%-15s %8d %10s %n", "Carrie", carrieTotal,
percent.format((double)carrieTotal / totalVotes));
System.out.format("%-15s %8d %10s %n", "TOTAL VOTES:", totalVotes,
"");
```

Notice:

- I changed `%-10s` → `%-15s` (so the first column has room for the longer label).
- Also used `%d` for integers instead of `%s` (clearer and avoids type confusion).

---

## Example Output

Candidate	Votes	Percentage
John	40	40%
Carrie	60	60%
TOTAL VOTES:	100	

Now everything lines up neatly.

---

👉 Do you want me to also show you how to add **separator lines** (like  
 -----) under the header so it looks more like a real results  
 table?```

**Dijiste:**

how do i make it so the final percentage isint rounded

**ChatGPT dijo:**

Good catch 👍 – by default, `NumberFormat.getPercentInstance()`  
 \*\*rounds to the nearest whole percent\*\*.  
 That's why you see ``40%`` instead of ``40.0%`` or ``40.25%``.

---

###  Solution: Control the decimal places

You can tell the formatter how many **\*\*fraction digits\*\*** to show:

```
```java
NumberFormat percent = NumberFormat.getPercentInstance();
percent.setMinimumFractionDigits(1); // always show at least 1 decimal
place
percent.setMaximumFractionDigits(2); // show up to 2 decimal places
```

So for example:

- $40 / 100 = 0.4 \rightarrow "40.0\%"$
- $403 / 1000 = 0.403 \rightarrow "40.3\%"$
- $407 / 1000 = 0.407 \rightarrow "40.7\%"$

Updated code snippet

```
NumberFormat percent = NumberFormat.getPercentInstance();
percent.setMinimumFractionDigits(1);
percent.setMaximumFractionDigits(2);

System.out.format("%-15s %8s %10s %n", "Candidate", "Votes",
"Percentage");
System.out.format("%-15s %8d %10s %n", "John", johnTotal,
percent.format((double)johnTotal / totalVotes));
System.out.format("%-15s %8d %10s %n", "Carrie", carrieTotal,
percent.format((double)carrieTotal / totalVotes));
System.out.format("%-15s %8d %10s %n", "TOTAL VOTES:", totalVotes,
"");
```

Example outputs

If John has 403 out of 1000 votes:

Candidate	Votes	Percentage
John	403	40.3%
Carrie	597	59.7%
TOTAL VOTES:	1000	

👉 Do you want the **TOTAL VOTES** row to also show the combined **100%** at the end, or should it stay blank?``