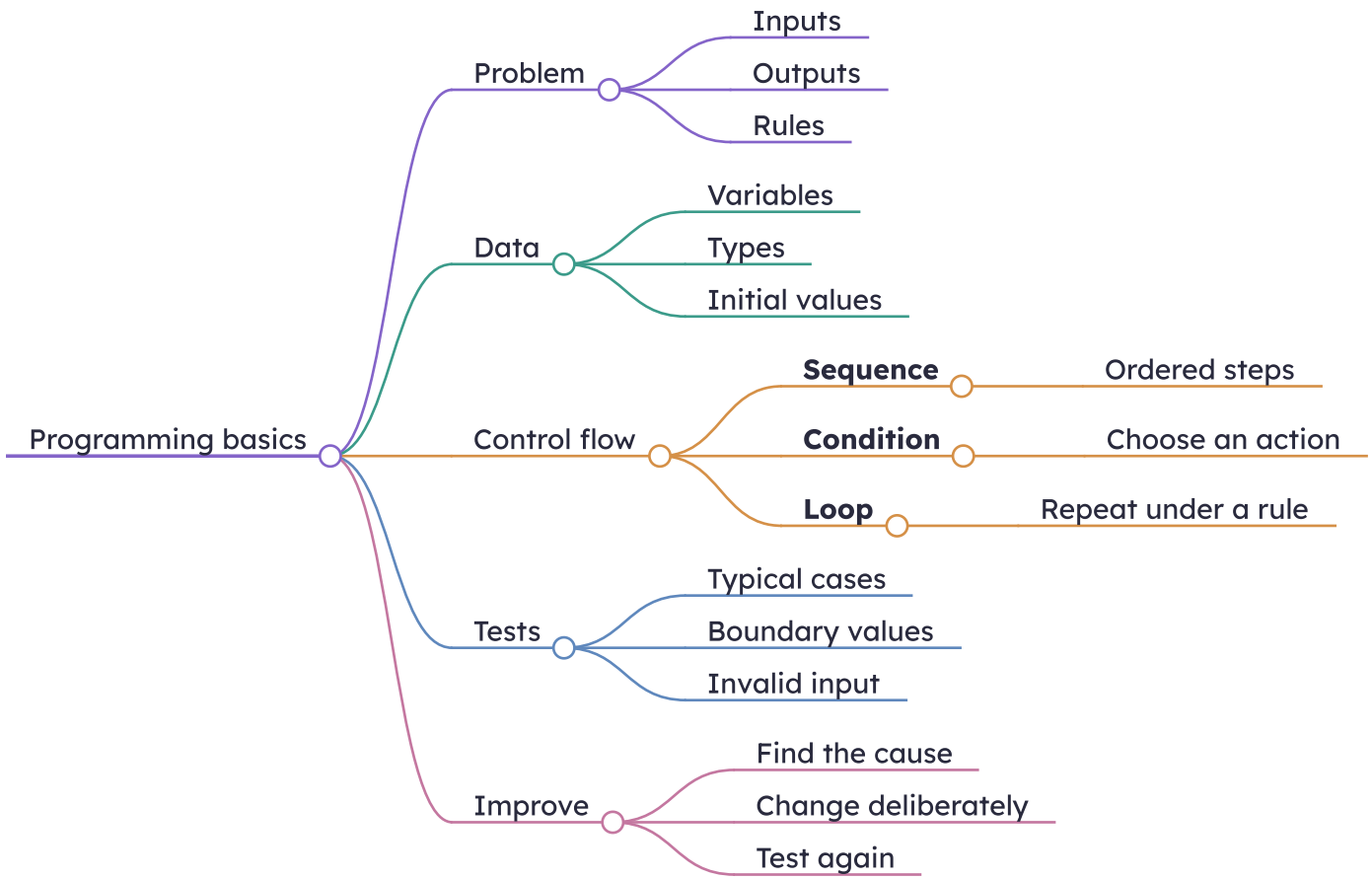
# Programming basics

Turn a problem into data, decisions, repetition, and tests.




Read one branch at a time. Then cover the details and explain them in your own words.

## Understand the topic

A program applies instructions to transform inputs into outputs. An algorithm describes the steps; code expresses them in a programming language. Variables hold values, conditions select actions, and loops repeat work. Tests should include normal cases, boundaries, and invalid input, with expected results written in advance.

### How the map is organized

Define the problem before choosing code structures. Keep testing alongside implementation.

> **Watch out for this**
> A program can run without errors and still give the wrong answer. Compare behavior with the requirements.

## Try it yourself

Design a checker for scores from 0 to 100. Reject invalid values before applying grade rules, then test 0, a boundary, 100, and 101.

### Two questions to answer

**Why test 101 for a 0–100 score range?**

An out-of-range case checks validation.

**What is an algorithm?**

An algorithm describes a process that code can implement.

### Sources

developer.mozilla.org
https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript

Original practice material compiled by Studera.ai. Explore at studera.ai/en/resources/mind-map-examples.