12. Create a new NotebookLM project titled: “My Chapter

Revision Notes.” 10-12-2025 a) Upload multiple sources (any 2) such as:

i. PDF notes

ii. Web articles

iii. Text copied into NotebookLM

b) Ask NotebookLM to:

i. Create a combined study guide using all

sources.

ii. Generate flashcards for quick revision.

iii. Create a concept map or explanation of

the topic. c) Manually check for:

i. Any incorrect facts

ii. Repeated information

iii. Missing important points

d) Attach 3 screenshots from NotebookLM:

i. Combined study guide

ii. Flashcards

iii. Concept map / explanation

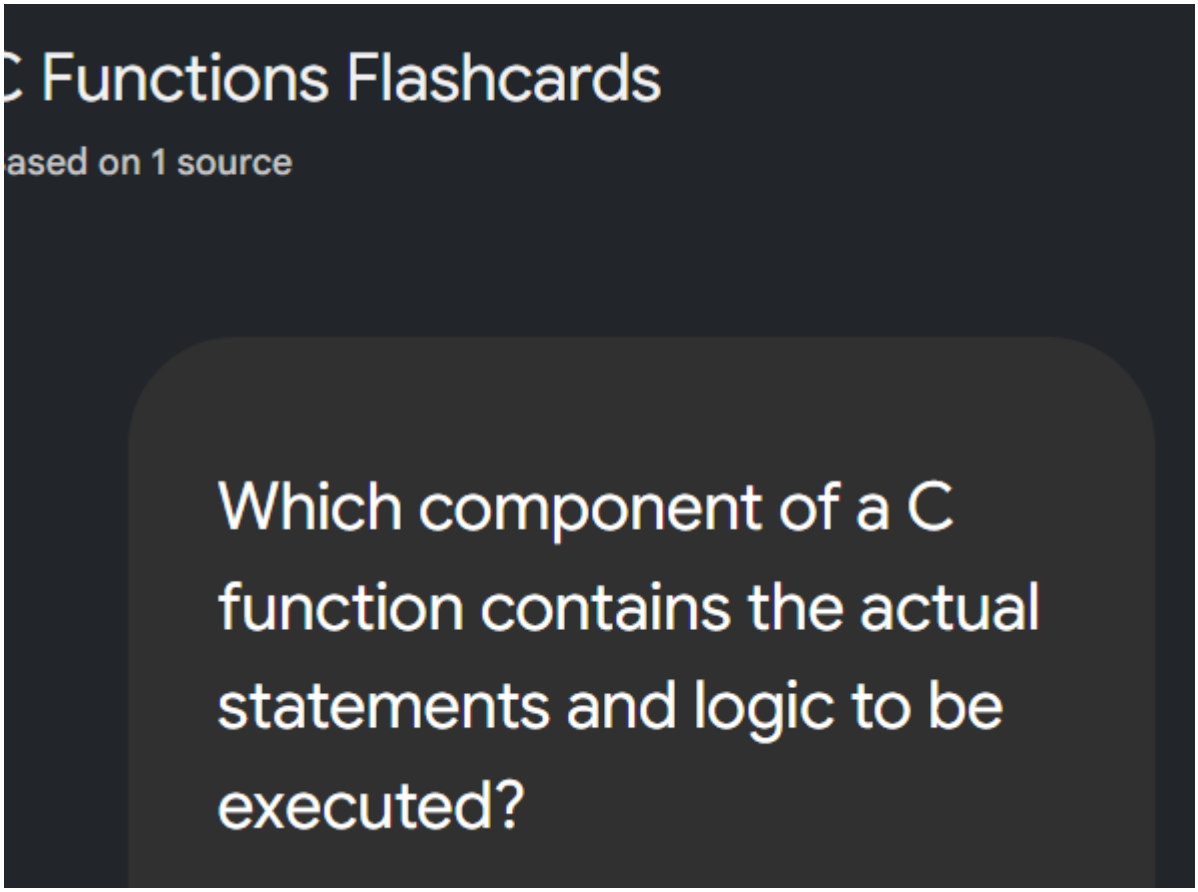
Short Study Guide: Functions in C

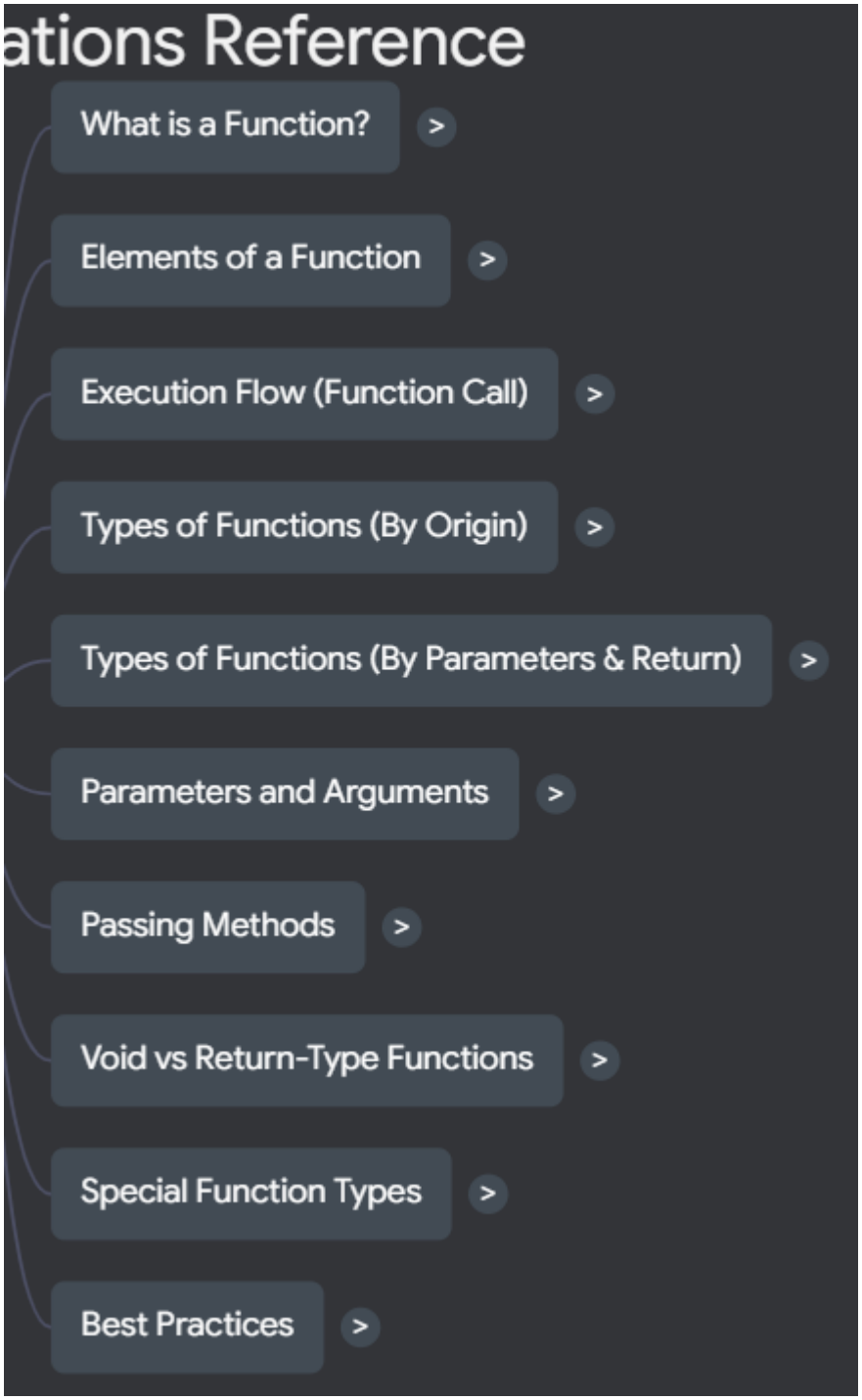
Functions are fundamental building blocks in C, acting as **named blocks of statements that perform a specific task** [1].

I. Purpose and Structure

Feature	Description	Citation
Why Use Functions?	Functions provide **modularity**, **reusability**, improved **readability**, **efficiency** (avoiding rewriting code), and **easy debugging** [1]. [1]	
Function Prototype	Declares the function's name, return type, and parameters to the compiler; ensures type checking; appears before `main()` [2]. [2]	
Function Definition	Contains the **actual logic** or statements that the function executes [2]. [	

Category	Parameters	Return Value	Use Case
1	❌ No	❌ No	Simple messages or actions (e.g., `void greet()`) [5] [6].
2	❌ No	✅ Yes	Returns constants or calculated values without external input [5] [6].
3	✅ Yes	❌ No	Takes input parameters but only prints results or performs actions (e.g., `void printSum(int a, int b)`) [6] [7].
4	✅ Yes	✅ Yes	Ideal for arithmetic and logic operations [6] [7].




ations Reference
What is a Function?
Elements of a Function
Execution Flow (Function Call)
Types of Functions (By Origin)
Types of Functions (By Parameters & Return)
Parameters and Arguments
Passing Methods
Void vs Return-Type Functions
Special Function Types
Best Practices