Rust Programming Projects

Detailed explanation of how the calculator app works.

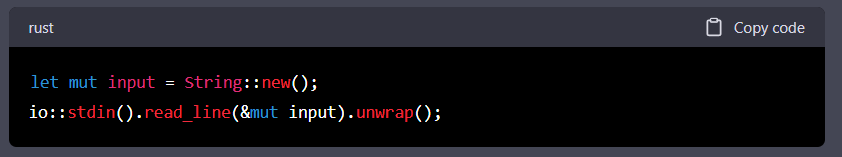
# Reading user input

The program starts by entering an infinite loop using the loop keyword. Inside the loop, it prompts the user for input using print! and io::stdout().flush().unwrap():



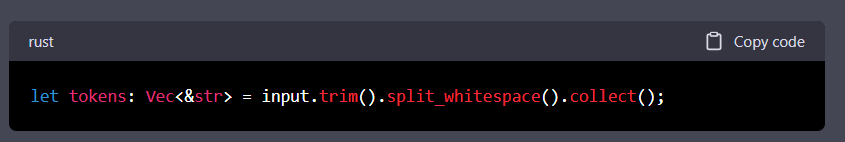
The flush method is used to ensure that the output is immediately displayed on the screen, rather than being buffered.

Next, the program reads a line of input from the user using io::stdin().read\_line(&mut input).unwrap();. The read\_line method reads a line of input (including the newline character) and appends it to the input variable:



# Parsing input

The program then splits the input into three tokens (left operand, operator, and right operand) using input.trim().split\_whitespace().collect();. The trim method removes any leading or trailing whitespace from the input, and split\_whitespace splits the input into tokens based on whitespace characters (spaces, tabs, and newlines). The collect method collects the tokens into a Vec<&str>:



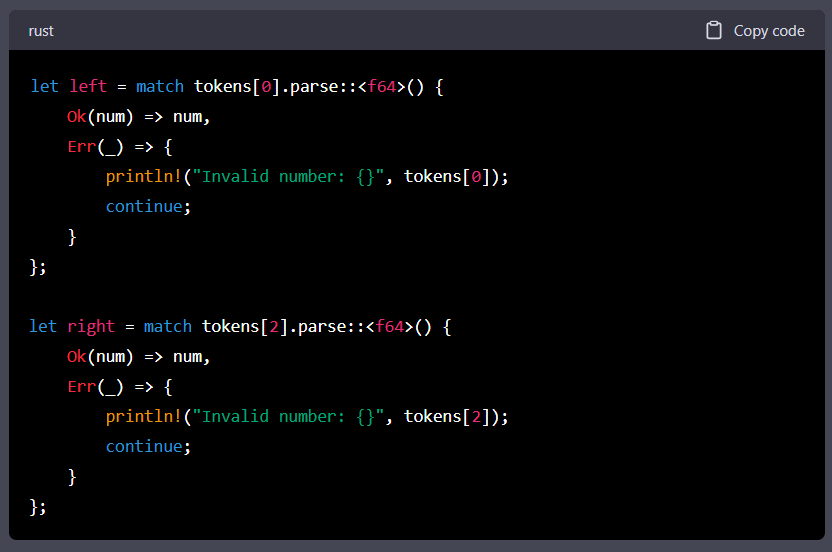
If there are not exactly three tokens, the program prints an error message and starts the loop again:

A picture containing graphical user interface

Description automatically generated

# Parsing operands

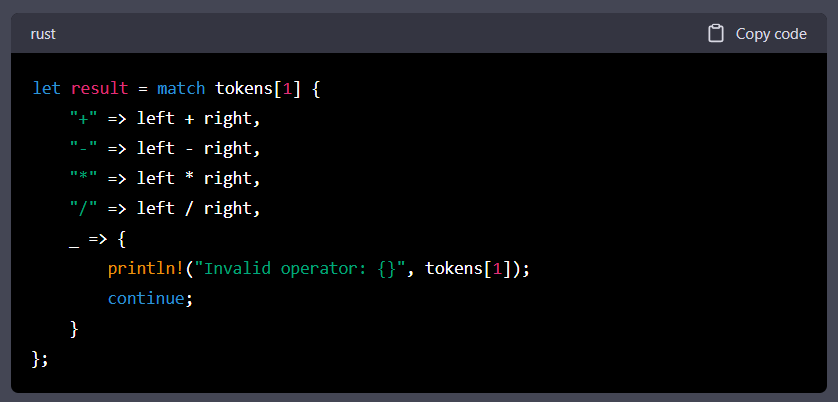
The program then attempts to parse the left and right operands as floating-point numbers using **parse::<f64>()**:



If the parsing is successful, the operands are stored in the left and right variables. If the parsing fails, the program prints an error message and starts the loop again.

# Performing the arithmetic operation

Finally, the program performs the arithmetic operation corresponding to the operator token using a match expression:



If the operator token is not one of the valid operators (+, -, \*, or /), the program prints an error message and starts the loop again. Otherwise, it performs the corresponding arithmetic operation and stores the result in the result variable.

# Displaying the result

Finally, the program prints the result to the console using println!("{}", result); and starts the loop again:

Background pattern

Description automatically generated