# Summary:

## Task-4:

In this task, I implemented a **recursive\_power(base, exponent)** function in C to compute base^exponent using a **recursive approach** instead of a loop. The function repeatedly calls itself until the exponent becomes zero, multiplying the base at each step.

I saved the code in a file named mypower.c, then compiled it using the **RISC-V cross-compiler** (riscv64-unknown-elf-gcc) with the -S flag to generate the assembly file mypower.s. This file contains the RISC-V assembly representation of the program.

Next, I assembled the .s file using riscv64-unknown-elf-as to create an object file (mypower.o), and then linked it using riscv64-unknown-elf-gcc to generate an executable ELF binary (mypower.elf).

To run the program, I used the **Spike RISC-V simulator** along with the **proxy kernel** (pk). Below are all the commands I used step by step:

**# Step 1**: Create and edit the C file

**nano mypower.c**

**# Step 2:** Compile to RISC-V assembly

**riscv64-unknown-elf-gcc -S mypower.c -o mypower.s**

**# Step 3:** Assemble to object file

**riscv64-unknown-elf-as mypower.s -o mypower.o**

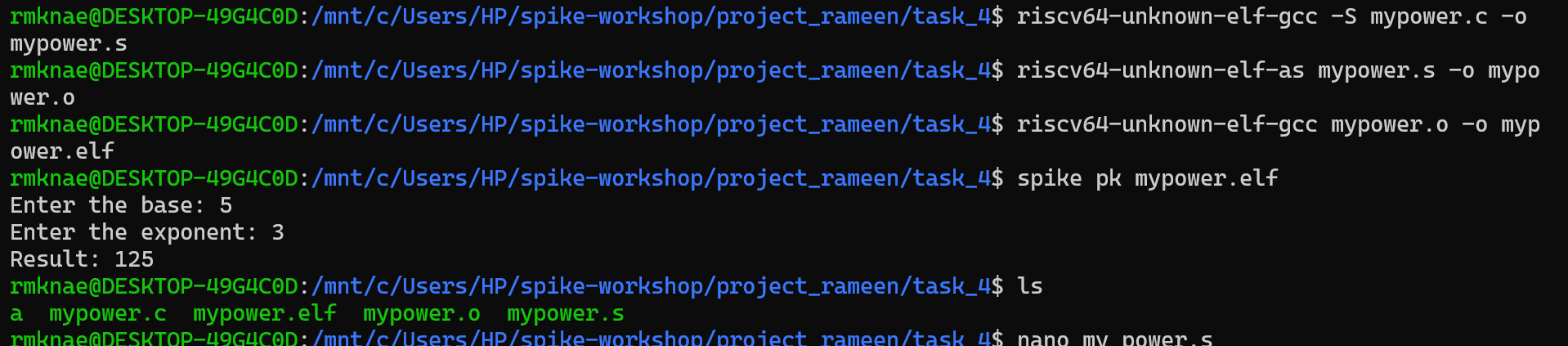
**# Step 4:** Link to generate ELF binary

**riscv64-unknown-elf-gcc mypower.o -o mypower.elf**

**# Step 5:** Run on Spike simulator

**spike pk mypower.elf**

**OUTPUT:**

****