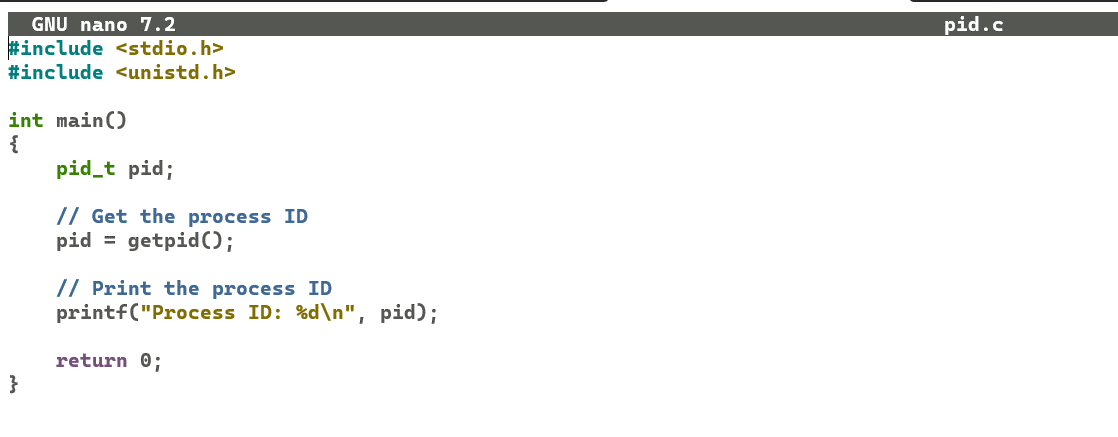
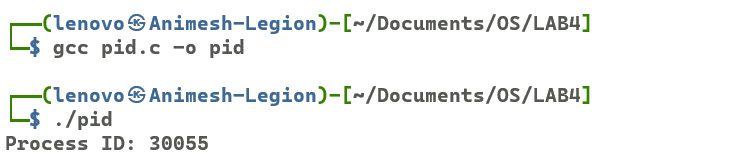
# Process

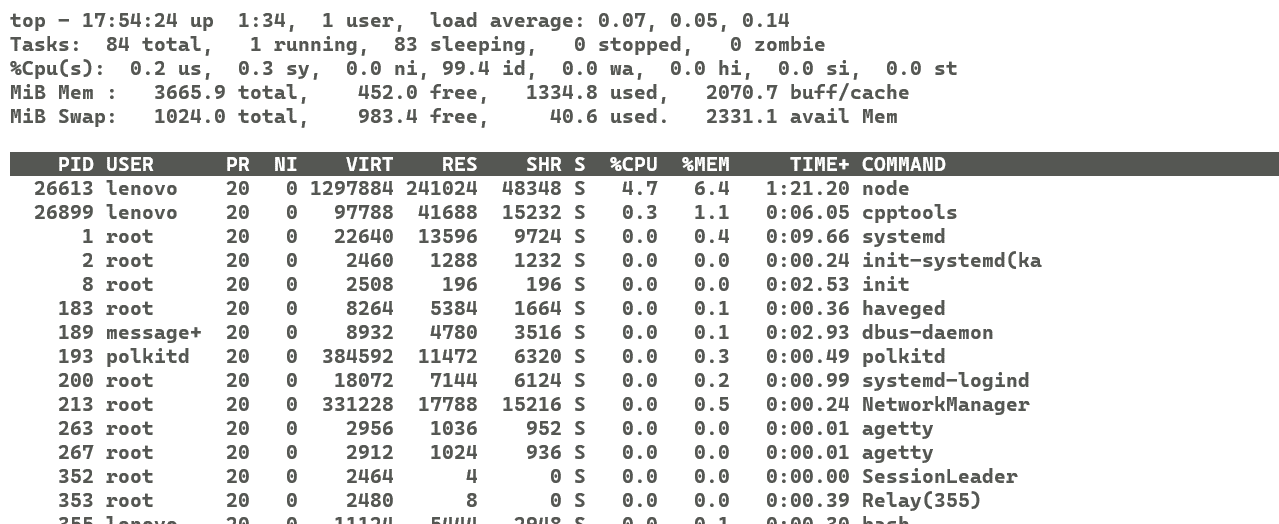
* 1. Program to print process ID - print-pid.c
     1. Creating the file

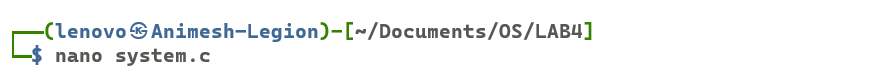
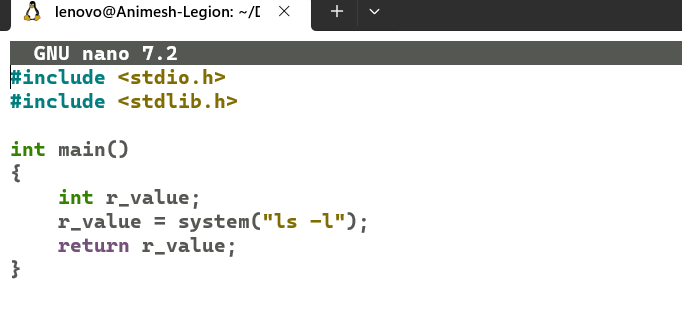


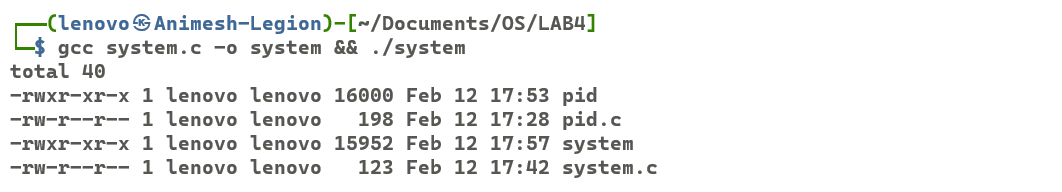
* + 1. Code:
    2. Output:



* + 1. Top:

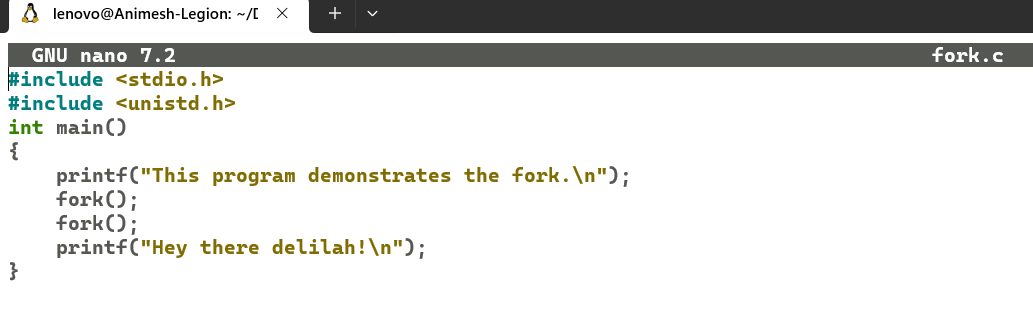
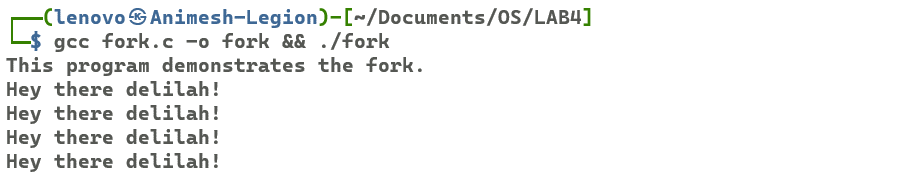
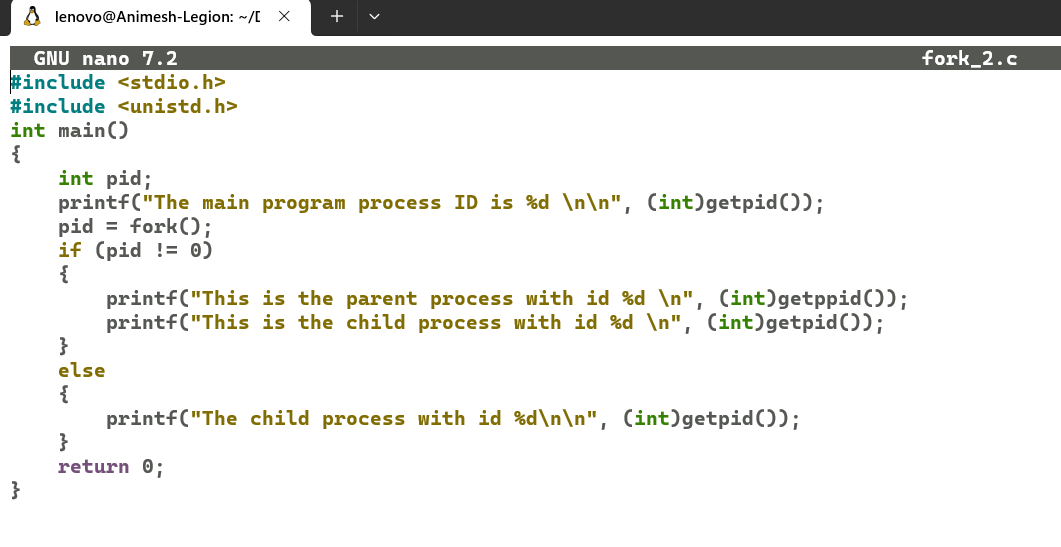
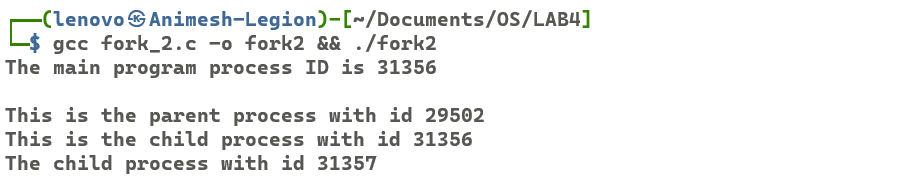
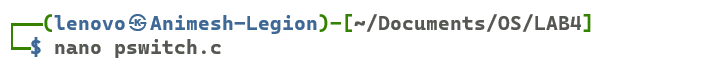
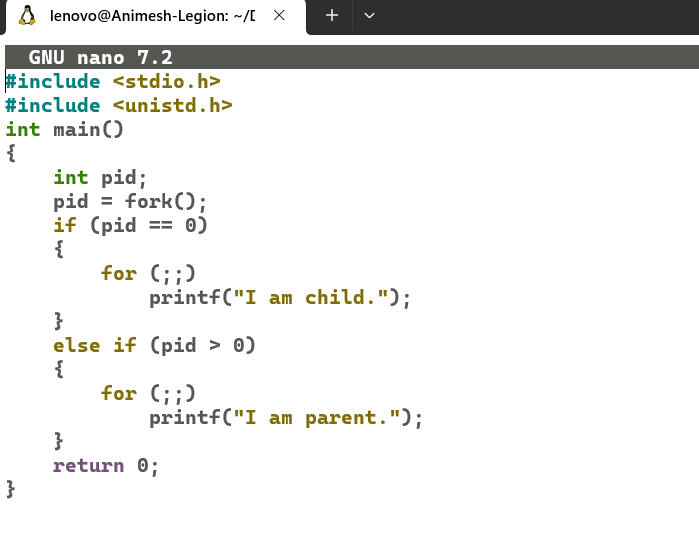
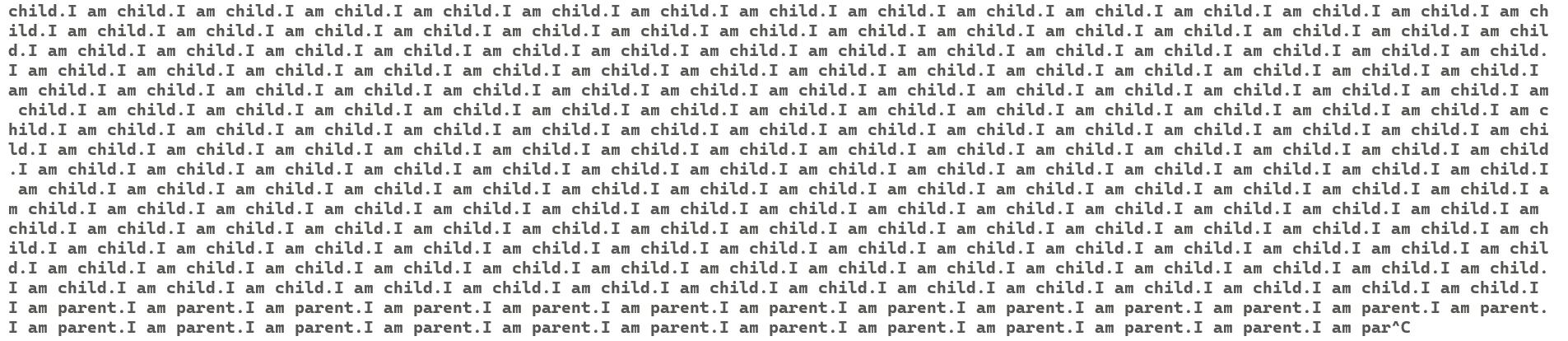
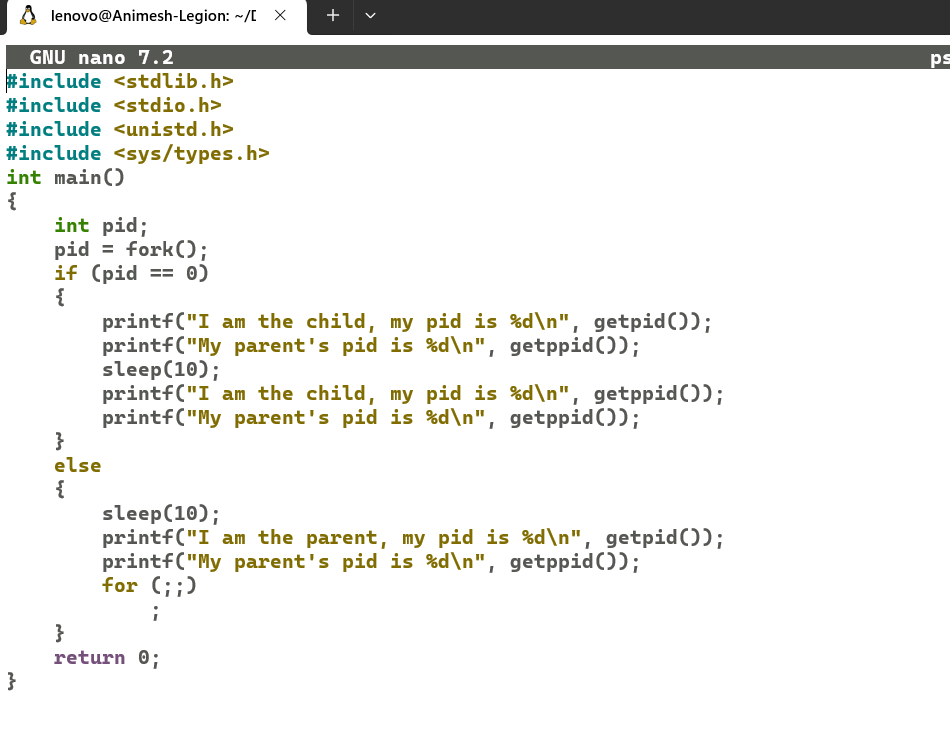
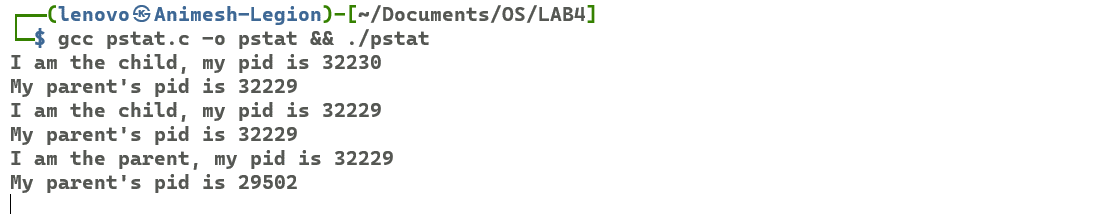
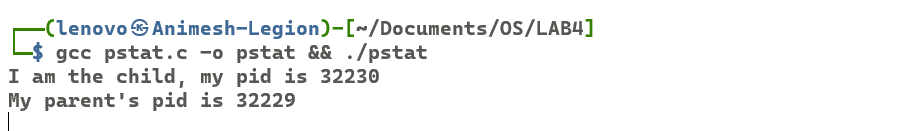


* 1. Program to create a process using system call - system.c
     1. Creating the file 
     2. Code: 
     3. Output:

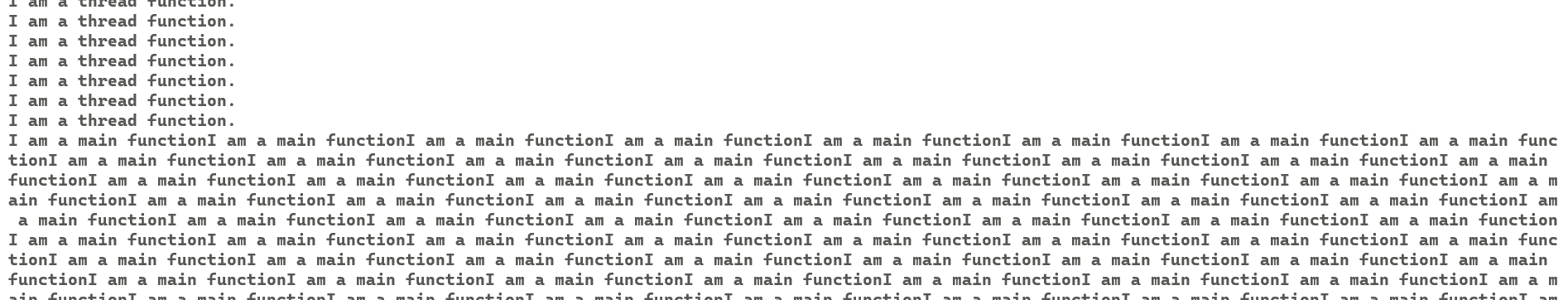
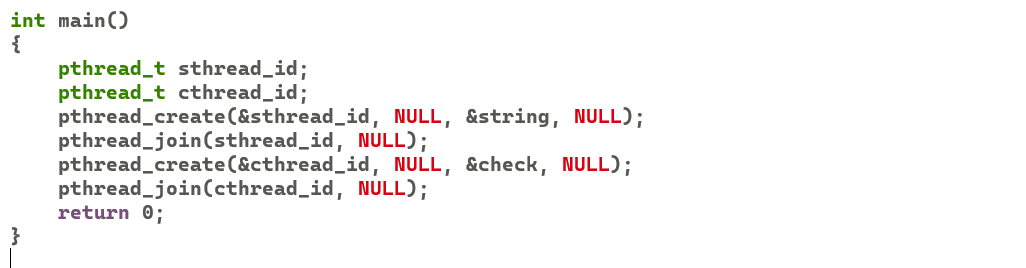
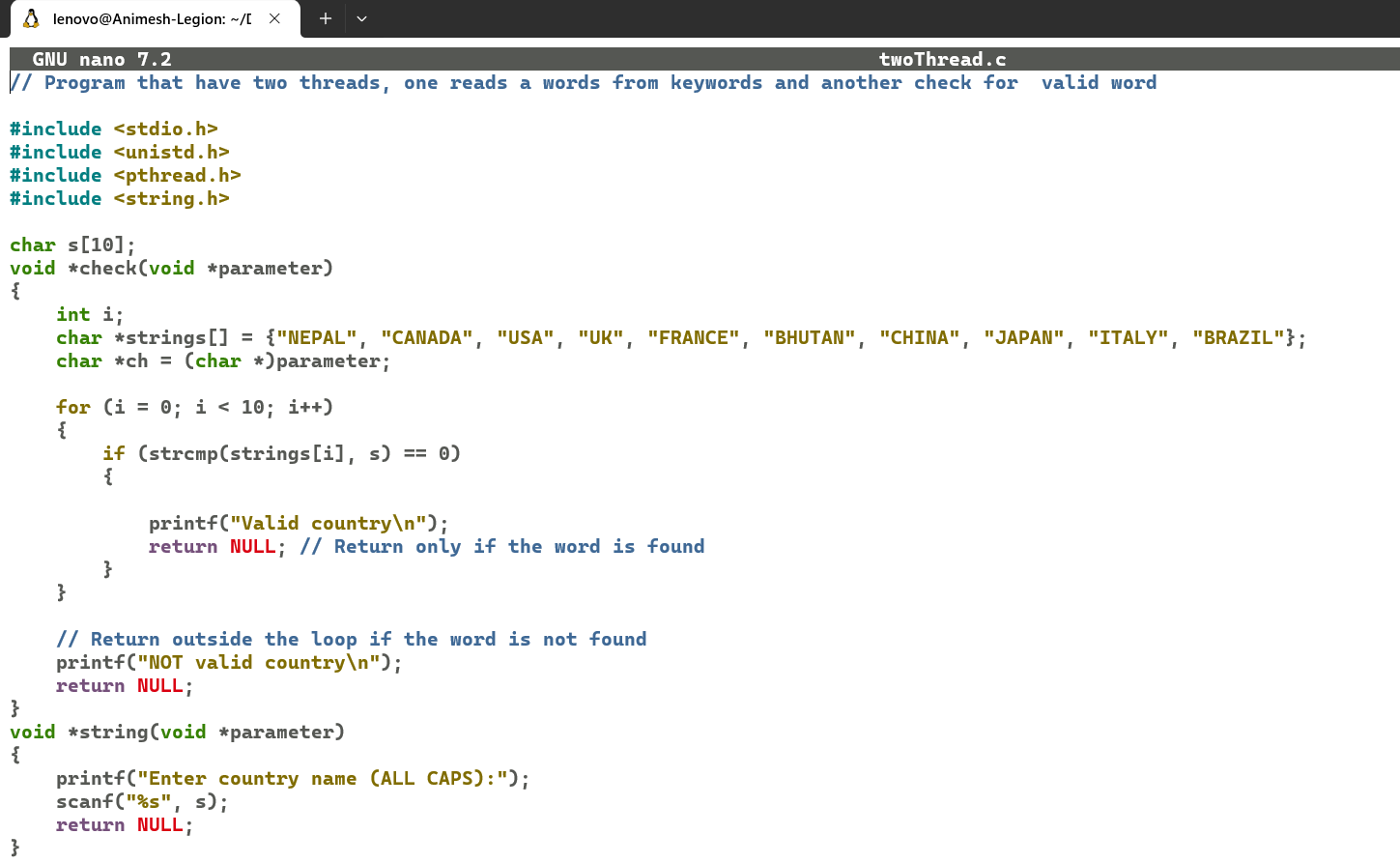
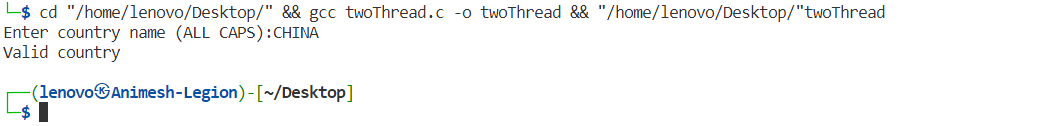
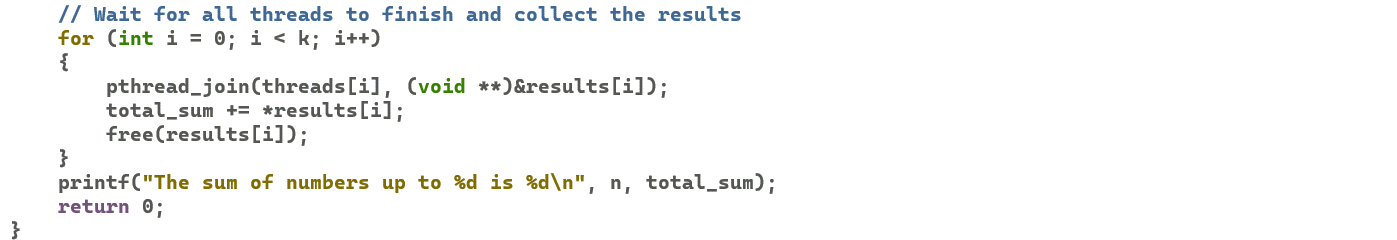
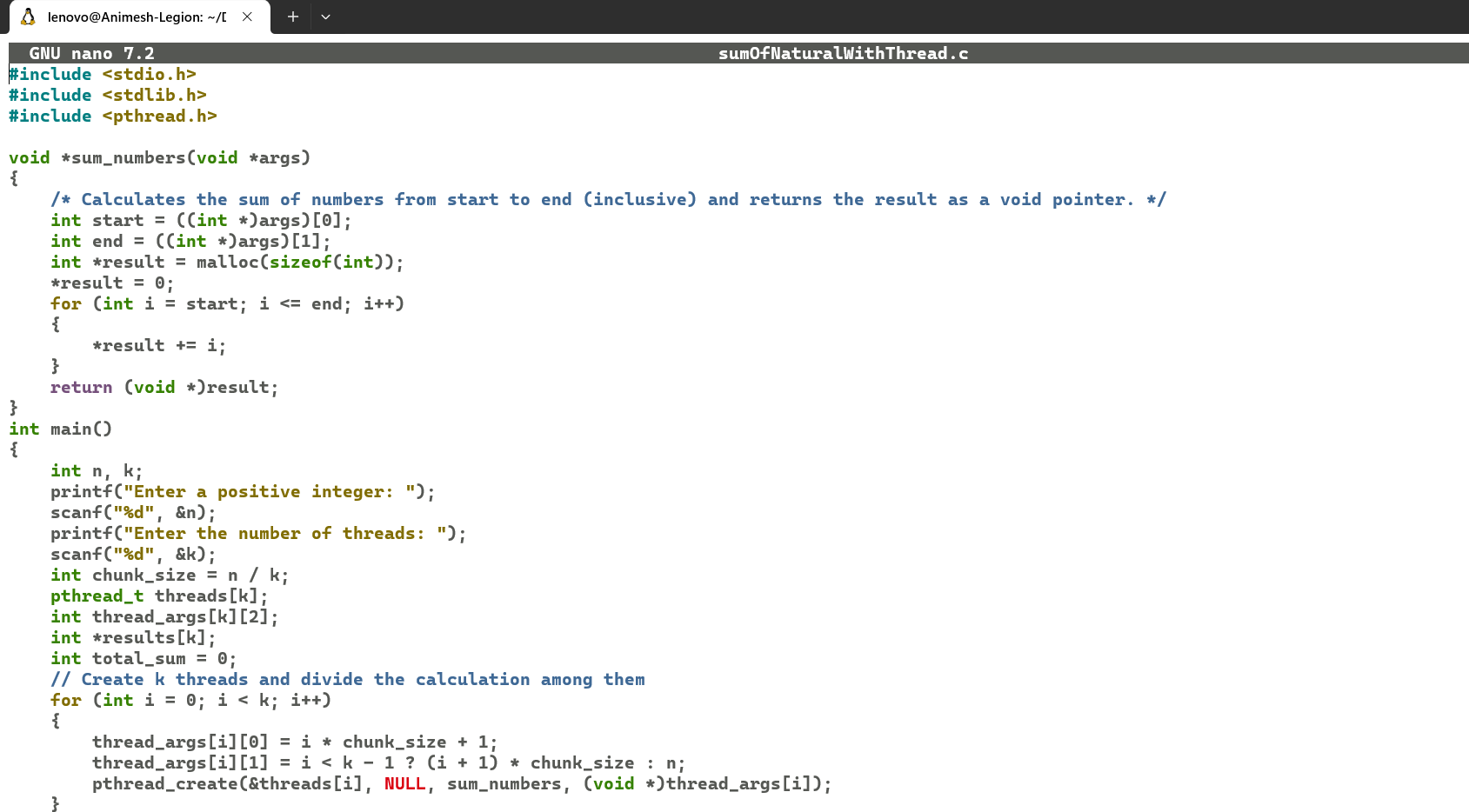


* 1. Program to create a process using fork - fork.c
     1. Creating the file:



* + 1. Code: 
    2. Output: 
  1. Program to create process using fork and print process id - fork\_2.c
     1. Creating the file: 
     2. Code: 
     3. Output: 
  2. Program to check the state and anlysize the output - pswitch.c
     1. Creating the file: 
     2. Code: 
     3. Output: 
  3. Execute the program and issue the command ps -el three times in every 10 second and anlysize their output -pstat.c
     1. Creating the file: 
     2. Code: 
     3. Output: 

# Threads

* 1. A simple implementation of thread
     1. Creating the file 
     2. Code: 
     3. Output: 
  2. Execute the ptoram and analyze the output; what changes will in your output when you remove last two line (pthread\_join), if any change, give reason.
     1. Creating the file
     2. Code:
     3. Output:
  3. Program that have two threads, one reads a word from keywords and another checks for valid word ( you can use your own word list, at least 10, to check validity)
     1. Creating the file 
     2.  Code: 
     3. Output: 
  4. Program using threads that prints sum of numbers up to given positive number n
     1. Creating the file
     2. Code: 
     3. Output:

