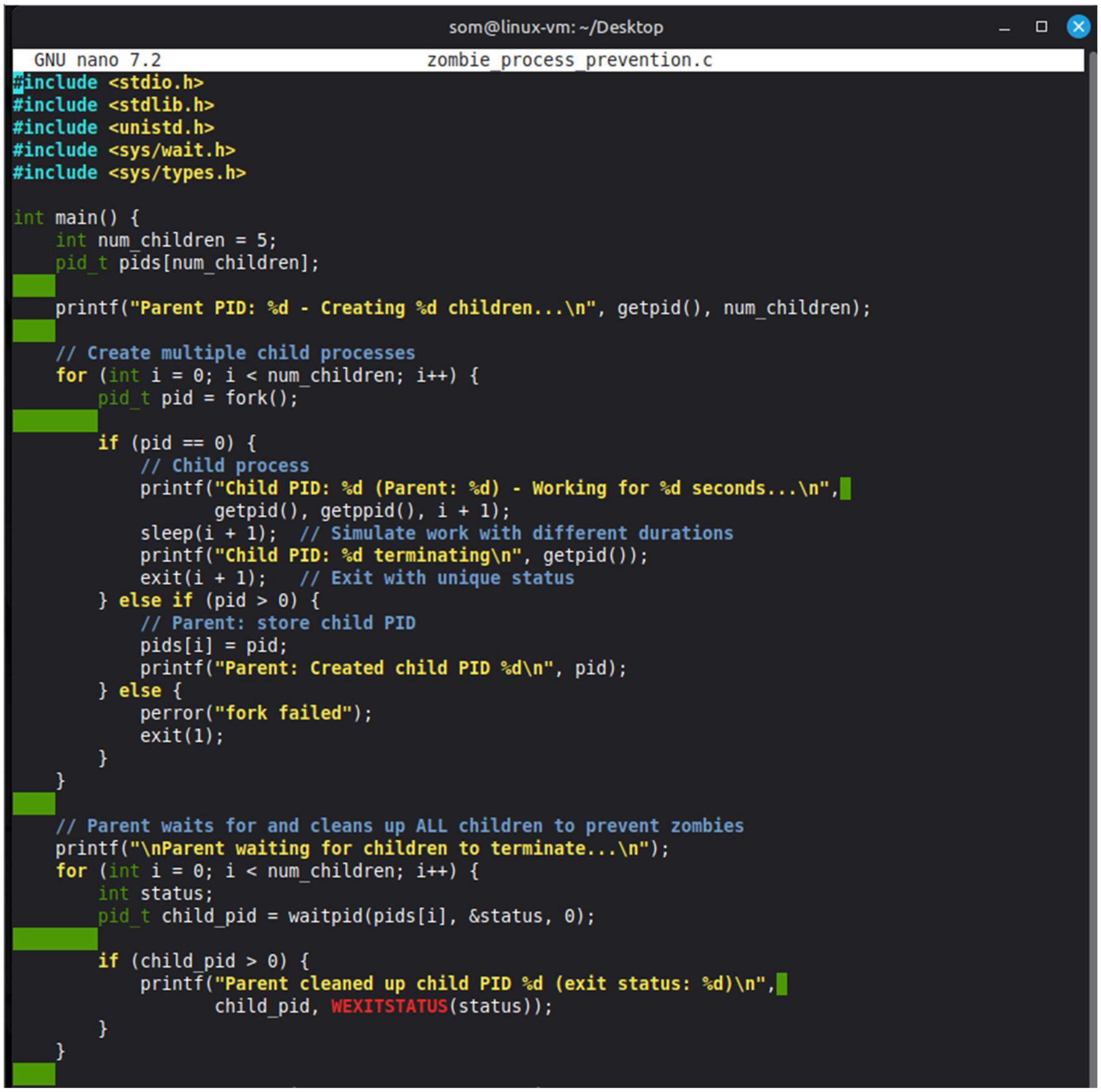
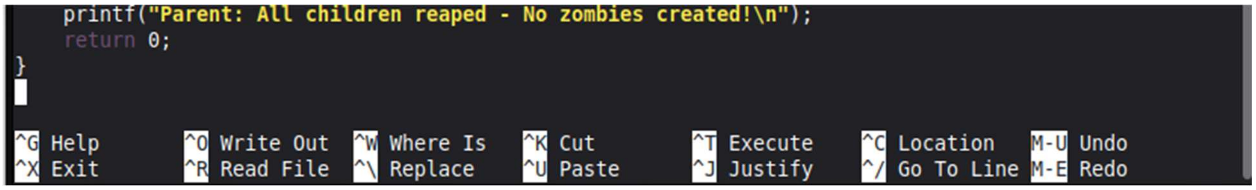
<u>Question 9 (2024eb03003):</u>

Please find screenshot of C program below and attaching **zombie_process_prevention.c** code to GitHub repository:



```
som@linux-vm: ~/Desktop
  GNU nano 7.2                     zombie_process_prevention.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>

int main() {
    int num_children = 5;
    pid_t pids[num_children];

    printf("Parent PID: %d - Creating %d children...\n", getpid(), num_children);

    // Create multiple child processes
    for (int i = 0; i < num_children; i++) {
        pid_t pid = fork();

        if (pid == 0) {
            // Child process
            printf("Child PID: %d (Parent: %d) - Working for %d seconds...\n",
                   getpid(), getppid(), i + 1);
            sleep(i + 1);  // Simulate work with different durations
            printf("Child PID: %d terminating\n", getpid());
            exit(i + 1);   // Exit with unique status
        } else if (pid > 0) {
            // Parent: store child PID
            pids[i] = pid;
            printf("Parent: Created child PID %d\n", pid);
        } else {
            perror("fork failed");
            exit(1);
        }
    }

    // Parent waits for and cleans up ALL children to prevent zombies
    printf("\nParent waiting for children to terminate...\n");
    for (int i = 0; i < num_children; i++) {
        int status;
        pid_t child_pid = waitpid(pids[i], &status, 0);

        if (child_pid > 0) {
            printf("Parent cleaned up child PID %d (exit status: %d)\n",
                   child_pid, WEXITSTATUS(status));
        }
    }

```



```
    printf("Parent: All children reaped - No zombies created!\n");
    return 0;
}

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo
```

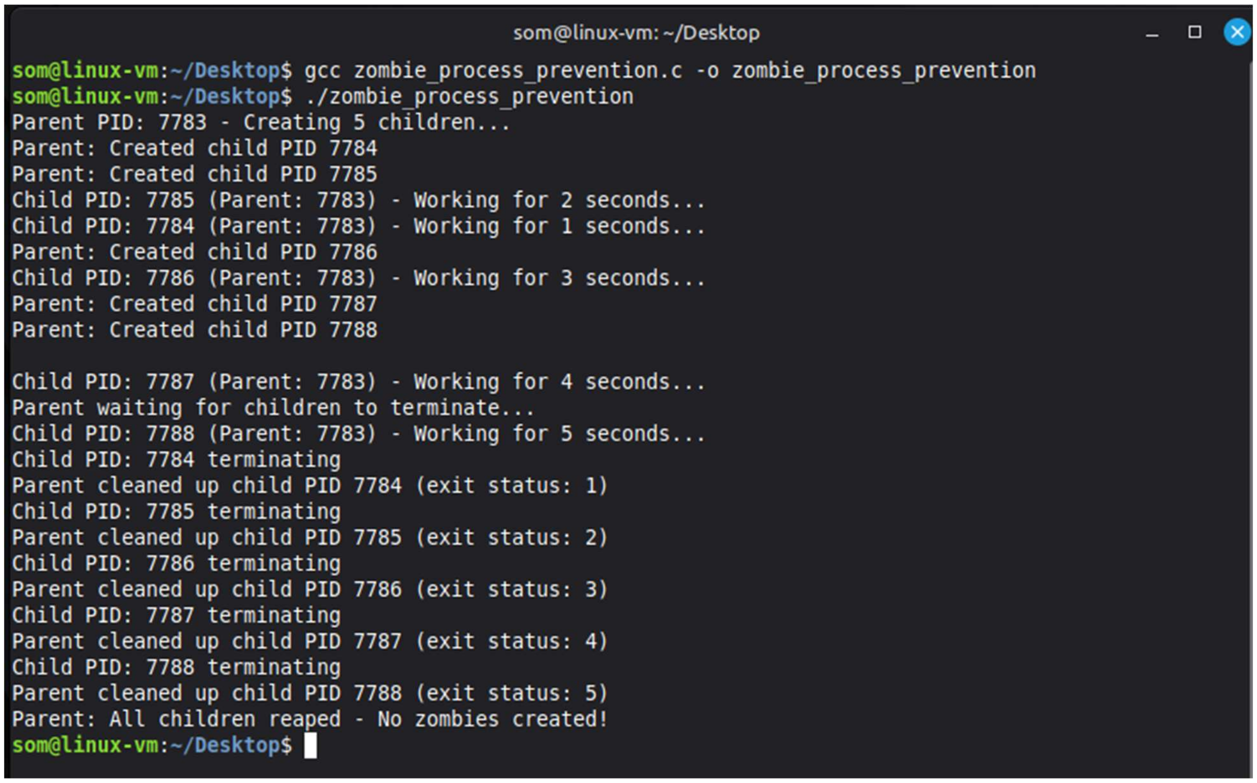
Testing the zombie_process_prevention.c code

Test Case:

Compile the c program by running below command

gcc zombie_process_prevention.c -o zombie_process_prevention

./zombie_prevent



```
som@linux-vm: ~/Desktop
som@linux-vm:~/Desktop$ gcc zombie_process_prevention.c -o zombie_process_prevention
som@linux-vm:~/Desktop$ ./zombie_process_prevention
Parent PID: 7783 - Creating 5 children...
Parent: Created child PID 7784
Parent: Created child PID 7785
Child PID: 7785 (Parent: 7783) - Working for 2 seconds...
Child PID: 7784 (Parent: 7783) - Working for 1 seconds...
Parent: Created child PID 7786
Child PID: 7786 (Parent: 7783) - Working for 3 seconds...
Parent: Created child PID 7787
Parent: Created child PID 7788

Child PID: 7787 (Parent: 7783) - Working for 4 seconds...
Parent waiting for children to terminate...
Child PID: 7788 (Parent: 7783) - Working for 5 seconds...
Child PID: 7784 terminating
Parent cleaned up child PID 7784 (exit status: 1)
Child PID: 7785 terminating
Parent cleaned up child PID 7785 (exit status: 2)
Child PID: 7786 terminating
Parent cleaned up child PID 7786 (exit status: 3)
Child PID: 7787 terminating
Parent cleaned up child PID 7787 (exit status: 4)
Child PID: 7788 terminating
Parent cleaned up child PID 7788 (exit status: 5)
Parent: All children reaped - No zombies created!
som@linux-vm:~/Desktop$
```

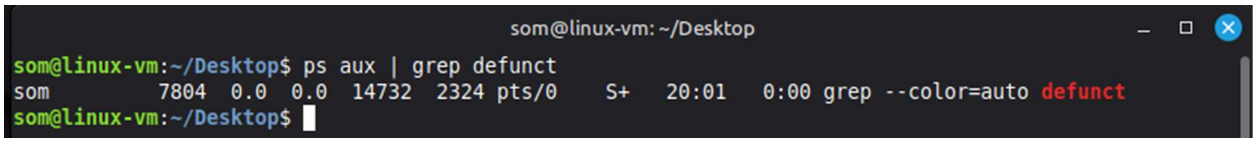
Validation Complete: The C program successfully:

- Created 5 children with fork()
- Parent reaped all children using waitpid()
- Printed each cleaned child PID
- Zero zombies remain

The above zombie prevention program works appropiately The parent properly cleaned up all terminated children before they could become zombies.

Verify no zombies:

We can see parent PID, 5 child PIDs created, each child works then terminates, parent prints "cleaned up child PID X" for each, and confirms "No zombies created!"



```
som@linux-vm:~/Desktop$ ps aux | grep defunct
som         7804  0.0  0.0  14732  2324 pts/0    S+   20:01   0:00 grep --color=auto defunct
som@linux-vm:~/Desktop$ 
```

ps aux | grep defunct output shows only the grep process itself (PID 7804), NOT a zombie process.

Explanation:

som   7804 0.0 0.0 14732 2324 pts/0   S+   20:01   0:00 grep --color=auto defunct

- This is grep matching itself (the [d]efunct trick prevents this)
- No <defunct> processes appear
- State S+ = sleeping (normal for grep)