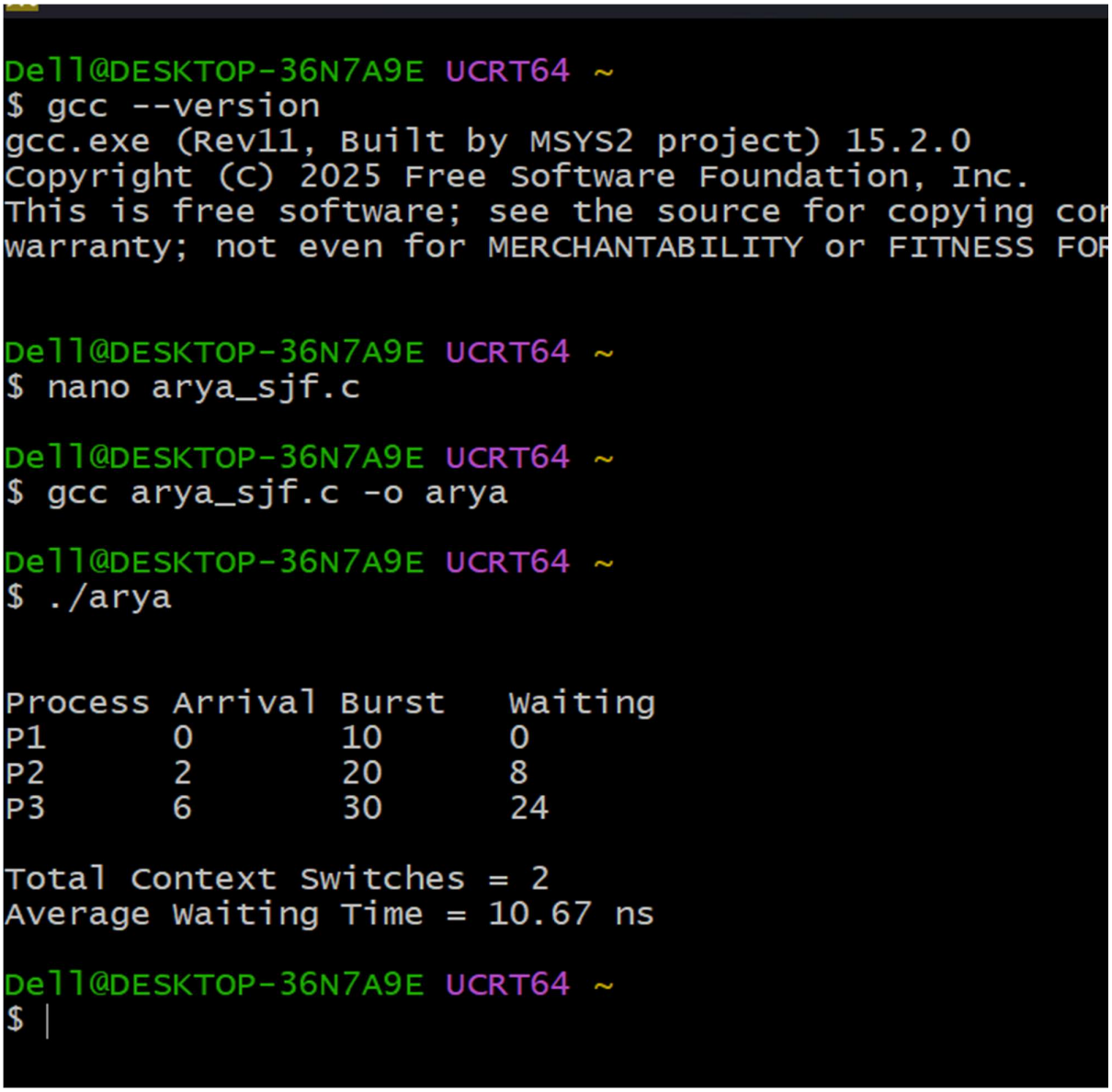
Practical 5 :

In an operating system three CPU-intensive processes are ready for execution, which require 10ns, 20ns and 30ns and arrival at times 0ns, 2ns and 6ns, respectively. Write a Program to calculate the total number of context switches needed if the operating system implements a shortest job first (preemptive) scheduling algorithm. Also calculate the average time for which the processes have to wait before getting the CPU.

```
  GNU nano 8.7                                                  arya_sjf.c
#include <stdio.h>

int main() {
    int arrival[3] = {0, 2, 6};
    int burst[3]   = {10, 20, 30};
    int completion[3];
    int waiting[3];
    int i;

    /* As per SRTF scheduling */
    completion[0] = 10;   // P1 finishes first
    completion[1] = 30;   // P2 finishes next
    completion[2] = 60;   // P3 finishes last

    for (i = 0; i < 3; i++) {
        waiting[i] = completion[i] - arrival[i] - burst[i];
    }

    float avg_wait = (waiting[0] + waiting[1] + waiting[2]) / 3.0;

    printf("\nProcess\tArrival\tBurst\tWaiting\n");
    for (i = 0; i < 3; i++) {
        printf("P%d\t%d\t%d\t%d\n", i+1, arrival[i], burst[i], waiting[i]);
    }

    printf("\nTotal Context Switches = 2");
    printf("\nAverage Waiting Time = %.2f ns\n", avg_wait);

    return 0;
}




                                                  [ Read 30 lines ]
^G Help         ^O Write Out    ^F Where Is     ^K Cut          ^T Execute      ^C Lo
```


Dell@DESKTOP-36N7A9E UCRT64 ~
$ gcc --version
gcc.exe (Rev11, Built by MSYS2 project) 15.2.0
Copyright (C) 2025 Free Software Foundation, Inc.
This is free software; see the source for copying cor
warranty; not even for MERCHANTABILITY or FITNESS FOR
Dell@DESKTOP-36N7A9E UCRT64 ~
$ nano arya_sjf.c
Dell@DESKTOP-36N7A9E UCRT64 ~
$ gcc arya_sjf.c -o arya
Dell@DESKTOP-36N7A9E UCRT64 ~
$ ./arya
Process Arrival Burst Waiting
P1 0 10 0
P2 2 20 8
P3 6 30 24
Total Context Switches = 2
Average Waiting Time = 10.67 ns
Dell@DESKTOP-36N7A9E UCRT64 ~
$