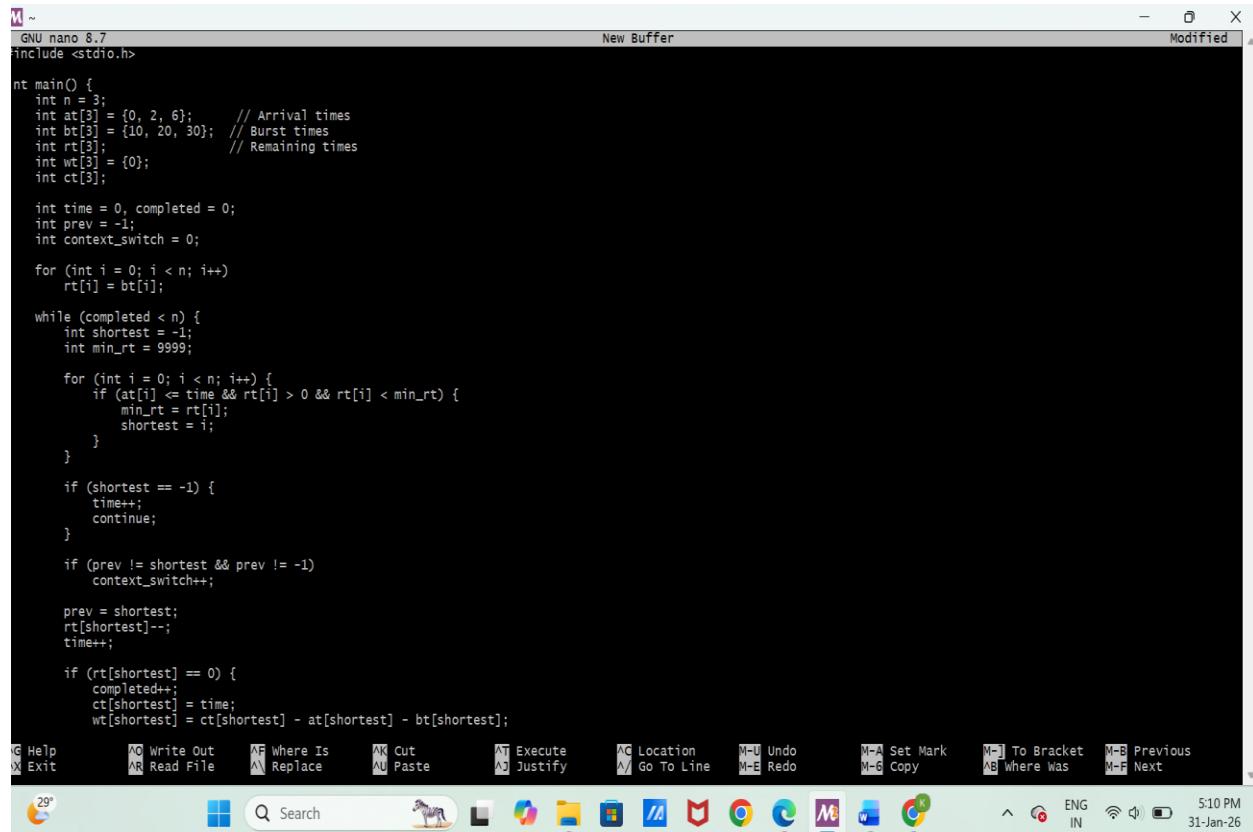


5.[Pracess management Based practical]

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.

CODE:



The screenshot shows a terminal window titled "GNU nano 8.7" with the file name "New Buffer". The code implements a Shortest Job First (SJF) scheduling algorithm to determine context switches and waiting times for three processes. The code uses arrays for arrival times (at), burst times (bt), remaining times (rt), waiting times (wt), and context switches (ct). It iterates through each process, finds the shortest remaining time, and updates the current time and context switch count accordingly. The terminal also displays a standard Linux-style menu bar and a bottom status bar with various icons and system information.

```
int main() {
    int n = 3;
    int at[3] = {0, 2, 6};      // Arrival times
    int bt[3] = {10, 20, 30};   // Burst times
    int rt[3];                 // Remaining times
    int wt[3] = {0};
    int ct[3];

    int time = 0, completed = 0;
    int prev = -1;
    int context_switch = 0;

    for (int i = 0; i < n; i++)
        rt[i] = bt[i];

    while (completed < n) {
        int shortest = -1;
        int min_rt = 9999;

        for (int i = 0; i < n; i++) {
            if (at[i] <= time && rt[i] > 0 && rt[i] < min_rt) {
                min_rt = rt[i];
                shortest = i;
            }
        }

        if (shortest == -1) {
            time++;
            continue;
        }

        if (prev != shortest && prev != -1)
            context_switch++;

        prev = shortest;
        rt[shortest]--;
        time++;

        if (rt[shortest] == 0) {
            completed++;
            ct[shortest] = time;
            wt[shortest] = ct[shortest] - at[shortest] - bt[shortest];
        }
    }
}
```

M ~

GNU nano 8.7

New Buffer

Modified

```
while (completed < n) {
    int shortest = -1;
    int min_rt = 9999;

    for (int i = 0; i < n; i++) {
        if (at[i] <= time && rt[i] > 0 && rt[i] < min_rt) {
            min_rt = rt[i];
            shortest = i;
        }
    }

    if (shortest == -1) {
        time++;
        continue;
    }

    if (prev != shortest && prev != -1)
        context_switch++;

    prev = shortest;
    rt[shortest]--;
    time++;

    if (rt[shortest] == 0) {
        completed++;
        ct[shortest] = time;
        wt[shortest] = ct[shortest] - at[shortest] - bt[shortest];
    }
}

float avg_wt = 0;
printf("\nProcess\tAt\tBt\tWt\n");
for (int i = 0; i < n; i++) {
    avg_wt += wt[i];
    printf("%d\t%d\t%d\t%d\n", i + 1, at[i], bt[i], wt[i]);
}

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

return 0;
}
```

AG Help

AE Write Out

AF Where Is

AK Cut

AU Paste

AT Execute

AJ Justify

AC Location

M-U Undo

M-A Set Mark

M-B To Bracket

M-G Where Was

M-B Previous

M-F Next

29

Search

Zebra

File

Folder

Image

Microsoft Word

Microsoft Excel

Microsoft Powerpoint

Google Chrome

Microsoft Edge

Microsoft Internet Explorer

Google Sheets

ENG IN

Wi-Fi

Speaker

510 PM

31-Jan-26

Output:

```
ASUS@Kalash-Laptop MSYS ~
$ nano sjf_preemptive.c

ASUS@Kalash-Laptop MSYS ~
$ gcc sjf_preemptive.c -o sjf

ASUS@Kalash-Laptop MSYS ~
$ ./sjf

Process   AT      BT      WT
P1        0       10      0
P2        2       20      8
P3        6       30     24

Total Context Switches = 2
Average Waiting Time = 10.67 ns

ASUS@Kalash-Laptop MSYS ~
```