

Practical 5

In an operating system three CPU-intensive processes are ready for execution, which require 10 ns, 20 ns and 30 ns and arrive at times 0 ns, 2 ns and 6 ns, 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:

```
hari.ai@Harryunlaptop: ~ + - FPS 1
GNU nano 7.2
#include <stdio.h>
#include <limits.h>

int main() {
    int n = 3;
    int burst[3] = {10, 20, 30};
    int arrival[3] = {0, 2, 6};
    int remaining[3];
    int waiting[3] = {0};
    int complete = 0, time = 0;
    int mmin = INT_MAX;
    int shortest = 0;
    int finish_time;
    int check = 0;
    int context_switch = 0;
    int prev = -1;

    for (int i = 0; i < n; i++)
        remaining[i] = burst[i];

    while (complete != n) {

        mmin = INT_MAX;
        check = 0;

        for (int j = 0; j < n; j++) {
            if ((arrival[j] <= time) &&
                (remaining[j] < mmin) &&
                (remaining[j] > 0)) {
                mmin = remaining[j];
                shortest = j;
                check = 1;
            }
        }

        if (check == 0) {
            time++;
        }
        else {
            remaining[shortest] -= 1;
            if (remaining[shortest] == 0)
                complete++;
            time++;
        }
    }

    printf("Completion Time: %d\n", time);
}

Wrote 74 lines ]
```

```
hari.ai@Harryfunlapcare: ~ + ~
GNU nano 7.2                                     sjf.c
}
if (prev != -1 && prev != shortest)
    context_switch++;

prev = shortest;
remaining[shortest]--;

minm = remaining[shortest];
if (minm == 0)
    minm = INT_MAX;

if (remaining[shortest] == 0) {
    completed++;
    finish_time = time + 1;

    waiting[shortest] =
        finish_time - burst[shortest] - arrival[shortest];

    if (waiting[shortest] < 0)
        waiting[shortest] = 0;
}

time++;
}

float total_wait = 0;
for (int i = 0; i < n; i++)
    total_wait += waiting[i];

printf("Total Context Switches: %d\n", context_switch);
printf("Average Waiting Time: %.2f\n", total_wait / n);

return 0;
}

```

NG Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-J To Bracket
EX Exit ^R Read File ^X Replace ^P Paste ^J Justify ^G Go To Line M-E Redo M-G Copy ^Q Where Was

ENG IN 69% 02:02 PM 17-02-2026

OUTPUT :

A screenshot of a terminal window titled "hari_aj@Harryfunlapcare ~". The window displays the following command-line session:

```
hari_aj@Harryfunlapcare:~$ nano sjf.c
hari_aj@Harryfunlapcare:~$ gcc sjf.c -o sjf
hari_aj@Harryfunlapcare:~$ ./sjf
Total Context Switches: 2
Average Waiting Time: 10.67
hari_aj@Harryfunlapcare:~$ |
```

The terminal is located on a Windows desktop, as evidenced by the taskbar icons at the bottom and the system tray indicators at the top right.