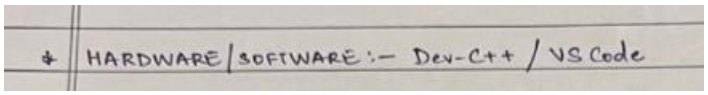
Name of Student: Vidhi Binwal	**PRN: 22070122249**
Semester: IV	**Year AY 23-24**
Subject Title: Operating Systems Lab	
EXPERIMENT No: 11	**Assignment No : 8**
TITLE: Scheduling Algorithm	**DoP : 30/3/2024**

Aim: Implement C program demonstrate **Shortest Remaining Job algorithm**

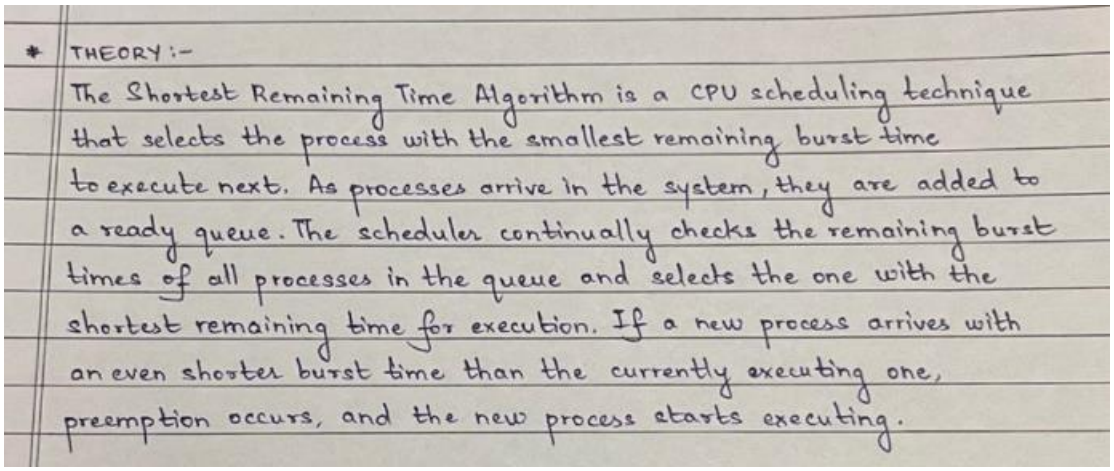
Learning Outcomes: *1. To understand the* **scheduling algorithm**

2. To Demonstrate the Shortest Remaining Job algorithm

***Hardware/Software*:**



Theory:



Algorithm:

1. Initialize: Set the current time to 0 and the set of ready processes to empty.

2. Arrival of Process: As processes arrive in the system, add them to the ready queue.

3. Selecting the Next Process:

   - If the ready queue is empty, the CPU remains idle.

   - If the ready queue is not empty, find the process with the smallest remaining burst time.

4. Executing the Process:

   - Execute the selected process for a predefined time quantum or until it completes its execution, whichever comes first.

5. Preemption:

   - If a new process arrives with a smaller remaining burst time than the currently executing process, preempt the current process and execute the newly arrived process.

6. Repeat:

   - Repeat steps 3-5 until all processes have completed their execution.

7. Completion:

   - When a process completes execution, remove it from the system.

8. Calculate Metrics:

   - Calculate performance metrics such as waiting time, turnaround time, and CPU utilization.

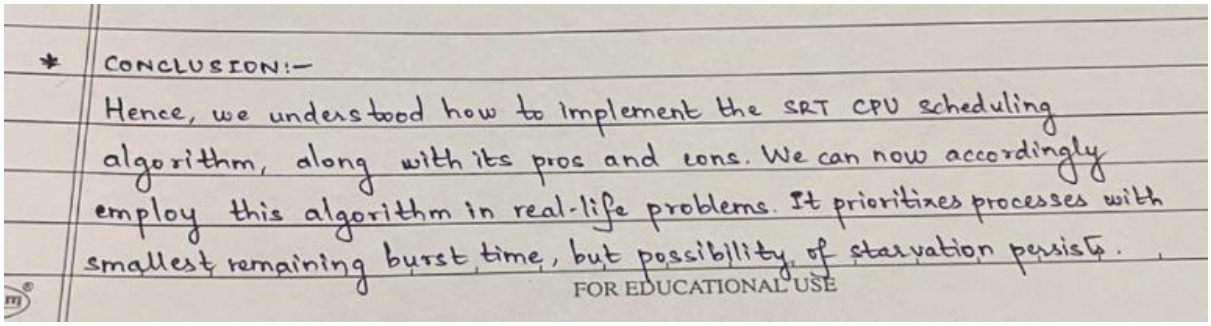
Program:

```c
#include <stdio.h>
#include <stdbool.h>

struct Process {
    int id;
    int arrival_time;
    int burst_time;
    int remaining_time;
    int start_time;
    int completion_time;
};

void srt(struct Process processes[], int n) {
```

```
    int current_time = 0;
    bool is_completed[n];
    int total_completed = 0;
    for (int i = 0; i < n; i++)
        is_completed[i] = false;

    while (total_completed < n) {
        int shortest_remaining = -1;
        int shortest_burst = 999999;
        for (int i = 0; i < n; i++) {
            if (!is_completed[i] && processes[i].arrival_time <= current_time
&&
                processes[i].burst_time < shortest_burst) {
                shortest_burst = processes[i].burst_time;
                shortest_remaining = i;
            }
        }
        if (shortest_remaining == -1) {
            current_time++;
            continue;
        }
        processes[shortest_remaining].remaining_time--;
        current_time++;

        if (processes[shortest_remaining].remaining_time == 0) {
            processes[shortest_remaining].completion_time = current_time;
            total_completed++;
            is_completed[shortest_remaining] = true;
        }
    }
}

int main() {
    int n;
    printf("Enter the number of processes: ");
    scanf("%d", &n);

    struct Process processes[n];
    for (int i = 0; i < n; i++) {
        printf("Enter arrival time and burst time for process %d: ", i + 1);
        scanf("%d %d", &processes[i].arrival_time, &processes[i].burst_time);
        processes[i].id = i + 1;
        processes[i].remaining_time = processes[i].burst_time;
    }

    printf("Shortest Remaining Time (SRT) scheduling:\n");
    srt(processes, n);
```

```
    int total_waiting_time = 0;
    int total_turnaround_time = 0;
    for (int i = 0; i < n; i++) {
        processes[i].start_time = processes[i].completion_time -
processes[i].burst_time;
        total_waiting_time += processes[i].start_time -
processes[i].arrival_time;
        total_turnaround_time += processes[i].completion_time -
processes[i].arrival_time;
    }
    double avg_waiting_time = (double) total_waiting_time / n;
    double avg_turnaround_time = (double) total_turnaround_time / n;

    printf("Average Waiting Time: %.2lf\n", avg_waiting_time);
    printf("Average Turnaround Time: %.2lf\n", avg_turnaround_time);

    return 0;
}
```

Conclusion:



Output:

```
PS C:\Users\Dell\AppData\Local\Temp> cd "C:\Users\Dell\AppD
Enter the number of processes: 4
Enter arrival time and burst time for process 1: 0 6
Enter arrival time and burst time for process 2: 2 4
Enter arrival time and burst time for process 3: 4 2
Enter arrival time and burst time for process 4: 6 8
Shortest Remaining Time (SRT) scheduling:
Average Waiting Time: 3.50
Average Turnaround Time: 8.50
PS C:\Users\Dell\AppData\Local\Temp>
```