6b) To implement the Shortest Job First (SJF) scheduling technique

Program code:

```
        scanf("%d", &bt[i]);
        p[i] = i + 1;  // process number
    }

    // Sort burst time and process number using Bubble Sort
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - i - 1; j++) {
            if (bt[j] > bt[j + 1]) {
                // Swap burst time
                temp = bt[j];
                bt[j] = bt[j + 1];
                bt[j + 1] = temp;
                // Swap process number
                temp = p[j];
                p[j] = p[j + 1];
                p[j + 1] = temp;
            }
        }
    }

    wt[0] = 0; // first process has no waiting time

    // Calculate waiting time
    for (i = 1; i < n; i++) {
        wt[i] = 0;
        for (j = 0; j < i; j++)
            wt[i] += bt[j];
        avg_wt += wt[i];
    }

    // Calculate turnaround time
    for (i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
        avg_tat += tat[i];
    }

    avg_wt /= n;
    avg_tat /= n;

    // Display results
    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (i = 0; i < n; i++) {
        printf("P[%d]\t%d\t\t%d\t\t%d\n", p[i], bt[i], wt[i], tat[i]);
    }

    printf("\nAverage Waiting Time: %.2f", avg_wt);
    printf("\nAverage Turnaround Time: %.2f\n", avg_tat);

    return 0;
}
```

Output:

```
Enter the number of processes: 4
Enter the burst time for each process:
P[1]: 6
P[2]: 8
P[3]: 7
P[4]: 3

Process Burst Time      Waiting Time    Turnaround Time
P[4]    3               0               3
P[1]    6               3               9
P[3]    7               9               16
P[2]    8               16              24

Average Waiting Time: 7.00
Average Turnaround Time: 13.00
```