CODE 4: CPU SCHEDULING SHORTEST JOB FIRST (SJF)

```
#include <stdio.h>


int main() {
    int n, i, j;
    float avgWT = 0, avgTAT = 0;


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


    int burstTime[n], waitingTime[n], turnaroundTime[n], process[n];


    // Input burst times
    printf("Enter burst times for each process:\n");
    for (i = 0; i < n; i++) {
        process[i] = i + 1;  // process IDs
        printf("P%d: ", i + 1);
        scanf("%d", &burstTime[i]);
    }


    // Sort processes by burst time (SJF)
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (burstTime[i] > burstTime[j]) {
                // swap burst times
                int temp = burstTime[i];
```

```
            burstTime[i] = burstTime[j];
            burstTime[j] = temp;

            // swap process IDs
            temp = process[i];
            process[i] = process[j];
            process[j] = temp;
        }
    }
}

// First process waiting time = 0
waitingTime[0] = 0;

// Calculate waiting times
for (i = 1; i < n; i++) {
    waitingTime[i] = waitingTime[i - 1] + burstTime[i - 1];
}

// Calculate turnaround times
for (i = 0; i < n; i++) {
    turnaroundTime[i] = waitingTime[i] + burstTime[i];
}

// Calculate averages
for (i = 0; i < n; i++) {
    avgWT += waitingTime[i];
    avgTAT += turnaroundTime[i];
```

```
    }
    avgWT /= n;
    avgTAT /= 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", process[i], burstTime[i], waitingTime[i],
turnaroundTime[i]);
    }

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

    return 0;
}
```