

**4. Construct a scheduling program with C that selects the waiting process with the smallest execution time to execute next.**

```
#include <stdio.h>
int main() {
    int n, bt[20], wt[20], tat[20], i, j;
    int total_wt = 0, total_tat = 0;
    int temp;

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

    printf("Enter burst time of each process:\n");
    for (i = 0; i < n; i++) {
        printf("P%d: ", i + 1);
        scanf("%d", &bt[i]);
    }

    // Sort burst times (SJF logic)
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (bt[i] > bt[j]) {
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;

            }
        }
    }

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

    // Calculate waiting time
    for (i = 1; i < n; i++)
        wt[i] = wt[i - 1] + bt[i - 1];

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

    // Calculate total times
    for (i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
    }
}
```

```

// Display results
printf("\nProcess\tBurst\tWaiting\tTurnaround\n");
for (i = 0; i < n; i++)
    printf("P%d\t%d\t%d\t%d\n", i + 1, bt[i], wt[i], tat[i]);

printf("\nAverage Waiting Time = %.2f", (float)total_wt / n);
printf("\nAverage Turnaround Time = %.2f\n", (float)total_tat / n);

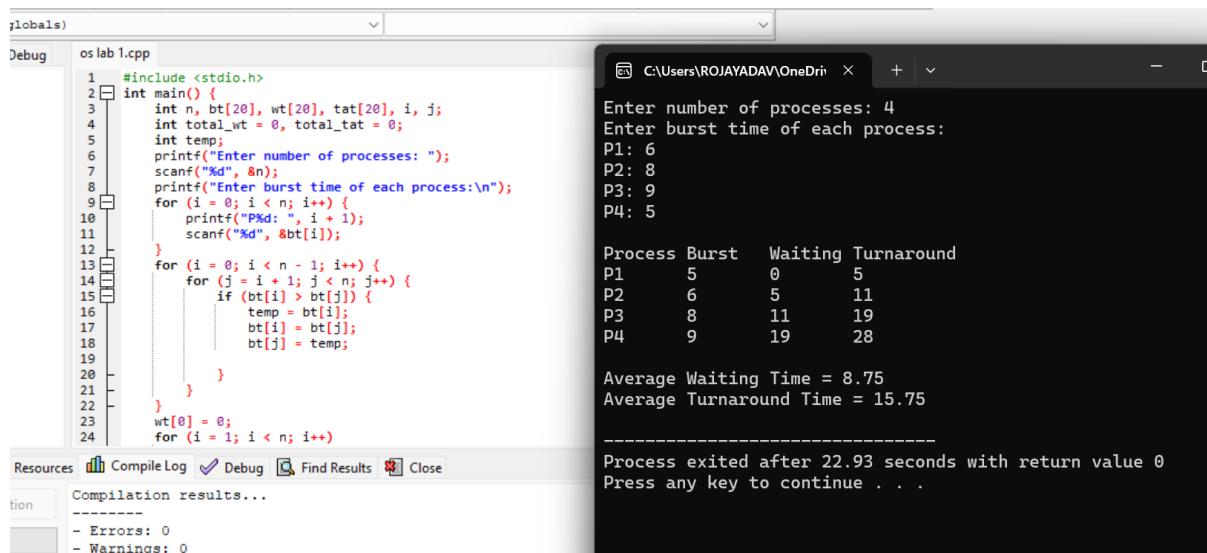
return 0;
}

```

### INPUT:

Enter number of processes: 4  
 Enter burst time of each process:  
 P1: 6  
 P2: 8  
 P3: 7  
 P4: 3

### OUTPUT:



```

#include <stdio.h>
int main() {
    int n, bt[20], wt[20], tat[20], i, j;
    int total_wt = 0, total_tat = 0;
    int temp;
    printf("Enter number of processes: ");
    scanf("%d", &n);
    printf("Enter burst time of each process:\n");
    for (i = 0; i < n; i++) {
        printf("P%d: ", i + 1);
        scanf("%d", &bt[i]);
    }
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (bt[i] > bt[j]) {
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;
            }
        }
    }
    wt[0] = 0;
    for (i = 1; i < n; i++)

```

Enter number of processes: 4  
 Enter burst time of each process:  
 P1: 6  
 P2: 8  
 P3: 9  
 P4: 5

	Process	Burst	Waiting	Turnaround
P1	5	0	5	
P2	6	5	11	
P3	8	11	19	
P4	9	19	28	

Average Waiting Time = 8.75  
 Average Turnaround Time = 15.75

---

Process exited after 22.93 seconds with return value 0  
 Press any key to continue . . .