1. **First Come First Serve (FCFS)**

```
#include <stdio.h>

int main() {
    int n, i;
    int bt[20], wt[20], tat[20];
    float avg_wt = 0, avg_tat = 0;

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

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

    wt[0] = 0;
    for(i = 1; i < n; i++) {
        wt[i] = wt[i-1] + bt[i-1];
    }

    for(i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
        avg_wt += wt[i];
        avg_tat += tat[i];
    }

    printf("\nProcess\tBT\tWT\tTAT\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", avg_wt/n);
    printf("\nAverage Turnaround Time: %.2f\n", avg_tat/n);

    return 0;
}
```

Sample Output

Enter number of processes: 3

Enter Burst Time for each process:
P1: 5
P2: 3
P3: 8

Process	BT	WT	TAT
P1	5	0	5
P2	3	5	8
P3	8	8	16

Average Waiting Time: 4.33
Average Turnaround Time: 9.67

2. **Shortest Job First (SJF) – Non-Preemptive**

```
#include <stdio.h>

int main() {
    int n, i, j, temp;
    int bt[20], p[20], wt[20], tat[20];
    float avg_wt = 0, avg_tat = 0;

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

    for(i = 0; i < n; i++) {
        printf("Enter Burst Time for P%d: ", i+1);
        scanf("%d", &bt[i]);
        p[i] = i + 1;
    }

    // Sorting by burst time
    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;
                temp = p[i]; p[i] = p[j]; p[j] = temp;
            }
        }
    }

    wt[0] = 0;
    for(i = 1; i < n; i++) {
        wt[i] = 0;
```

```
        for(j = 0; j < i; j++) wt[i] += bt[j];
        avg_wt += wt[i];
    }

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

    printf("\nProcess\tBT\tWT\tTAT\n");
    for(i = 0; i < n; i++) {
        printf("P%d\t%d\t%d\t%d\n", p[i], bt[i], wt[i], tat[i]);
    }

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

    return 0;
}
```

Sample output

```
Enter number of processes: 3
Enter Burst Time for P1: 6
Enter Burst Time for P2: 8
Enter Burst Time for P3: 7

Process BT  WT  TAT
P1      6   0   6
P3      7   6   13
P2      8   13  21

Average Waiting Time: 6.33
Average Turnaround Time: 13.33
```

3. Round Robin Scheduling

```
#include <stdio.h>

int main() {
    int n, i, tq, time = 0;
    int bt[10], rem_bt[10], wt[10] = {0}, tat[10];
    int done;

    printf("Enter number of processes: ");
```

```
scanf("%d", &n);

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

printf("Enter Time Quantum: ");
scanf("%d", &tq);

do {
   done = 1;
   for(i = 0; i < n; i++) {
      if(rem_bt[i] > 0) {
         done = 0;
         if(rem_bt[i] > tq) {
            time += tq;
            rem_bt[i] -= tq;
         } else {
            time += rem_bt[i];
            wt[i] = time - bt[i];
            rem_bt[i] = 0;
         }
      }
   }
} while(!done);

float avg_wt = 0, avg_tat = 0;
for(i = 0; i < n; i++) {
   tat[i] = bt[i] + wt[i];
   avg_wt += wt[i];
   avg_tat += tat[i];
}

printf("\nProcess\tBT\tWT\tTAT\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", avg_wt/n);
printf("\nAverage Turnaround Time: %.2f\n", avg_tat/n);
```

```
    return 0;
}
```

Sample output

```
Enter number of processes: 3
Enter Burst Time for each process:
P1: 10
P2: 5
P3: 8
Enter Time Quantum: 3

Process BT  WT  TAT
P1      10  13  23
P2      5   9   14
P3      8   14  22

Average Waiting Time: 12.00
Average Turnaround Time: 19.67
```