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

**✅ C Code:**

#include <stdio.h>

int main() {

int n, i;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], wt[n], tat[n];

float avg\_wt = 0, avg\_tat = 0;

printf("Enter burst times:\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] = wt[i] + bt[i];

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", i+1, bt[i], wt[i], tat[i]);

avg\_wt += wt[i];

avg\_tat += tat[i];

}

printf("\nAverage Waiting Time: %.2f", avg\_wt / n);

printf("\nAverage Turnaround Time: %.2f\n", avg\_tat / n);

return 0;

}

**🧾 Sample Output:**

less

CopyEdit

Enter number of processes: 3

Enter burst times:

P[1]: 5

P[2]: 8

P[3]: 12

Process Burst Time Waiting Time Turnaround Time

P[1] 5 0 5

P[2] 8 5 13

P[3] 12 13 25

Average Waiting Time: 6.00

Average Turnaround Time: 14.33

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

**✅ C Code:**

#include <stdio.h>

int main() {

int n, i, j;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], p[n], wt[n], tat[n];

float avg\_wt = 0, avg\_tat = 0;

printf("Enter burst times:\n");

for (i = 0; i < n; i++) {

printf("P[%d]: ", i+1);

scanf("%d", &bt[i]);

p[i] = i+1;

}

// Sort by burst time

for (i = 0; i < n-1; i++) {

for (j = i+1; j < n; j++) {

if (bt[i] > bt[j]) {

int 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] = wt[i-1] + bt[i-1];

for (i = 0; i < n; i++)

tat[i] = wt[i] + bt[i];

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]);

avg\_wt += wt[i];

avg\_tat += tat[i];

}

printf("\nAverage Waiting Time: %.2f", avg\_wt / n);

printf("\nAverage Turnaround Time: %.2f\n", avg\_tat / n);

return 0;

}

**🧾 Sample Output:**

less

CopyEdit

Enter number of processes: 3

Enter burst times:

P[1]: 6

P[2]: 2

P[3]: 8

Process Burst Time Waiting Time Turnaround Time

P[2] 2 0 2

P[1] 6 2 8

P[3] 8 8 16

Average Waiting Time: 3.33

Average Turnaround Time: 8.67

**3. Round Robin Scheduling**

**✅ C Code:**

#include <stdio.h>

int main() {

int i, n, tq, time = 0;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], rt[n], wt[n], tat[n];

printf("Enter burst times:\n");

for (i = 0; i < n; i++) {

printf("P[%d]: ", i+1);

scanf("%d", &bt[i]);

rt[i] = bt[i];

}

printf("Enter time quantum: ");

scanf("%d", &tq);

int done;

do {

done = 1;

for (i = 0; i < n; i++) {

if (rt[i] > 0) {

done = 0;

if (rt[i] > tq) {

time += tq;

rt[i] -= tq;

} else {

time += rt[i];

wt[i] = time - bt[i];

rt[i] = 0;

}

}

}

} while (!done);

float avg\_wt = 0, avg\_tat = 0;

printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");

for (i = 0; i < n; i++) {

tat[i] = bt[i] + wt[i];

printf("P[%d]\t%d\t\t%d\t\t%d\n", i+1, bt[i], wt[i], tat[i]);

avg\_wt += wt[i];

avg\_tat += 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 times:

P[1]: 10

P[2]: 5

P[3]: 8

Enter time quantum: 2

Process Burst Time Waiting Time Turnaround Time

P[1] 10 13 23

P[2] 5 6 11

P[3] 8 11 19

Average Waiting Time: 10.00

Average Turnaround Time: 17.67