* **Implementation of Round robin:**

#include <stdio.h>

struct process {

char name[10];

int burst;

int remaining;

int waiting;

};

int main() {

int n, quantum;

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

scanf("%d", &n);

struct process p[n];

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

printf("Enter the name of process %d: ", i + 1);

scanf("%s", p[i].name);

printf("Enter the burst time of process %d: ", i + 1);

scanf("%d", &p[i].burst);

p[i].remaining = p[i].burst;

p[i].waiting = 0;

}

printf("Enter the time quantum: ");

scanf("%d", &quantum);

int time = 0;

int done;

printf("\n---Process execution details---\n");

do {

done = 1;

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

if (p[i].remaining > 0) {

done = 0;

if (p[i].remaining > quantum) {

printf("Process %s runs from %d to %d\n", p[i].name, time, time + quantum);

time = time + quantum;

p[i].remaining = p[i].remaining - quantum;

} else {

printf("Process %s runs from %d to %d (completes)\n", p[i].name, time, time + p[i].remaining);

time = time + p[i].remaining;

p[i].waiting = time - p[i].burst;

p[i].remaining = 0;}} }

} while (!done);

printf("\n---Final process table---\n");

printf("Process Burst Waiting\n");

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

printf("%s %d %d\n", p[i].name, p[i].burst, p[i].waiting);

}

int totalwaiting = 0;

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

totalwaiting = totalwaiting + p[i].waiting; }

float avg = (float)totalwaiting / n;

printf("\nAverage waiting time is : %f\n", avg);

return 0;}

* **Ouput:**

