Ex. No.: 6d)
Date 21/3/25

ROUND ROBIN SCHEDULING

Aim:

To implement the Round Robin (RR) scheduling technique

Algorithm:

1. Declare the structure and its elements.
2. Get number of processes and Time quantum as input from the user.
3. Read the process name, arrival time and burst time
4. Create an array **rem_bt[]** to keep track of remaining burst time of processes which is initially copy of bt[] (burst times array)
5. Create another array **wt[]** to store waiting times of processes. Initialize this array as 0. 6. Initialize time : t = 0
7. Keep traversing the all processes while all processes are not done. Do following for i'th process if it is not done yet.
   a- If rem_bt[i] > quantum
   (i) t = t + quantum
   (ii) bt_rem[i] -= quantum;
   b- Else // Last cycle for this process
   (i) t = t + bt_rem[i];
   (ii) wt[i] = t - bt[i]
   (iii) bt_rem[i] = 0; // This process is over
8. Calculate the waiting time and turnaround time for each process.
9. Calculate the average waiting time and average turnaround time.
10. Display the results.

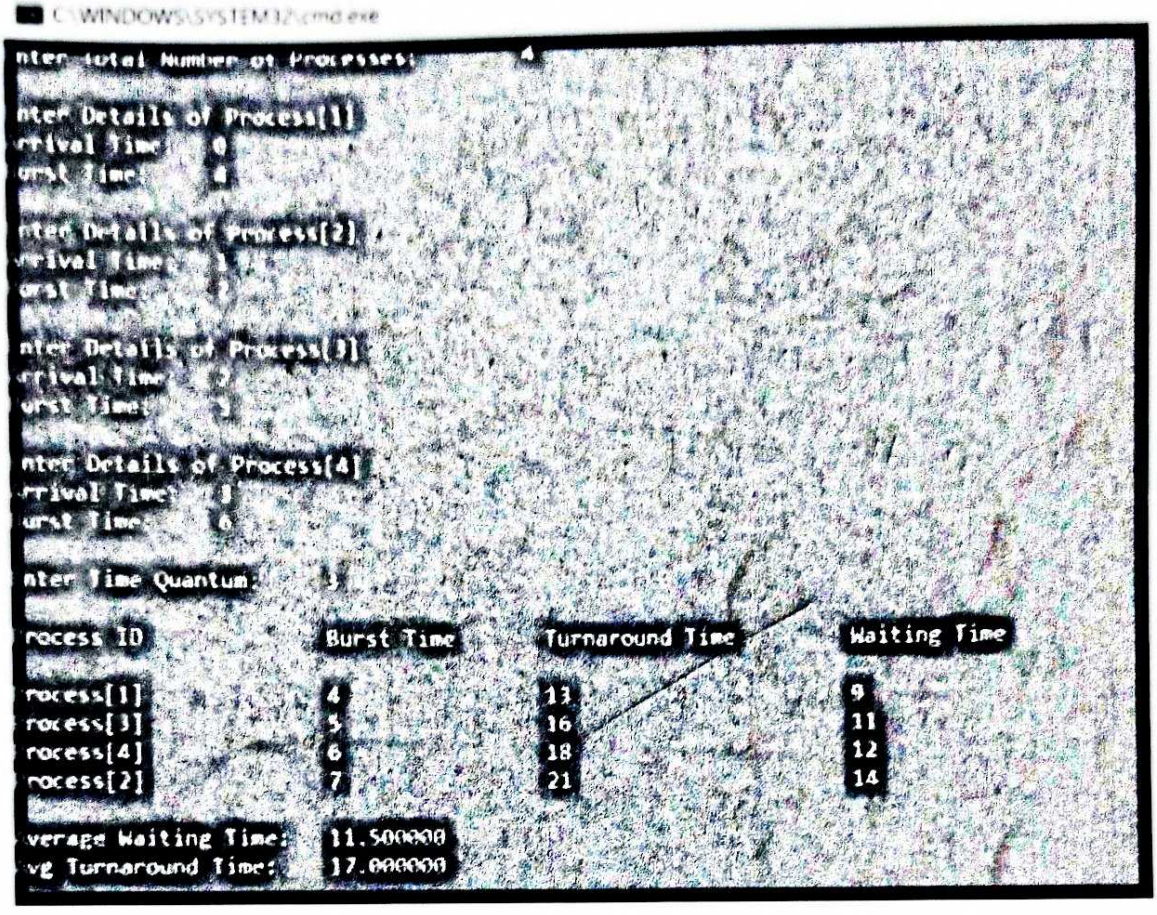
Program Code:

```
# include <stdio.h>
int main()
{
  int n, i, tq, time=0, remain, flag=0;
  printf("Enter the number of process:");
  scanf("%d", &n);
  int at[n], bt[n], rem_bt[n], wt[n], tat[n];
  float total_wt=0, total_tat=0;
printf("Enter the arrival time and burst time for
        each process:\n");
```

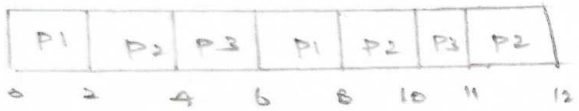
```
for (i=0 ; i<n ; i++) {
    printf(" Process [%d] .\n Arrival time": i);
    scanf ("%d", &at[i]);
    print ("Burst time");
    scanf ("%d", & bt[i];
printf ("Enter the time quantum:");
  scanf ("%d", &tq);
   remain = n;
printf ("\n Process Execution order:\n");
while (remain > 0) {
    for (i=0; i<n; i++) {
        if (rem_bt [i] > 0) {
            printf (p%d, i);
            if (rembt [i] > tq) {
              time += tq;
              rem_bt [i] -= tq;
            }
            else {
              time += rem_bt [i];
             wt [i] = time - bt [i] - at [i];
             tat [i] = time - at [i];
             rem_bt [i] = 0;
             remain--;
            }
        }
    }
}
```

```
printf("\n Process \t Arrival time \t Burst time \t turn around
                                        time \n");

for(i=0;i<n;i++) {
    printf("%d \t %d \t %d \t %d \n", i,
            at[i], bt[i], wt[i] tat[i]);

    total_wt += wt[i];
    total_tat += tat[i];
}
printf("\n Average Waiting time %.2f \n", total_wt/n);
printf("\n Average turn around time %.2\n", total tat/n);

}
```

Sample Output:



Gantt chart



Output

Process	Burst time	Waiting time	Turnaround time
1	4	4	8
2	5	7	12
3	3	8	11

Average tat = 10.33 ms

Average wt = 6.33 ms

Result:

Thus the round robin scheduling have been executed successfully.