```c
#include<stdio.h>

void round_robin(int processes[], int n, int burst_time[], int arrival_time[], int time_quantum)
{
  int remaining_time[n];
  for (int i = 0; i < n; i++)
    remaining_time[i] = burst_time[i];

  int current_time = 0;
  while (1)
  {
    int all_completed = 1;
    for (int i = 0; i < n; i++)
    {
      if (arrival_time[i] <= current_time && remaining_time[i] > 0)
      {
        all_completed = 0;

        if (remaining_time[i] > time_quantum)
        {
          current_time += time_quantum;
          remaining_time[i] -= time_quantum;
          printf("Executing process %d at time %d\n", processes[i], current_time);
        }
        else
        {
          current_time += remaining_time[i];
          remaining_time[i] = 0;
          printf("Executing process %d at time %d\n", processes[i], current_time);
        }
      }
    }

    if (all_completed)
      break;

    // If no process is available at current time, move to the next time slot
    if (all_completed && current_time < arrival_time[n - 1])
      current_time = arrival_time[n - 1];
  }
}

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

  int processes[n];
  int burst_time[n];
  int arrival_time[n];
  int time_quantum;
  printf("Enter the burst time and arrival time for each process:\n");
  for (int i = 0; i < n; i++)
  {
    printf("Process %d:\n", i + 1);
    printf("Burst Time: ");
```

```
        scanf("%d", &burst_time[i]);
        printf("Arrival Time: ");
        scanf("%d", &arrival_time[i]);
        processes[i] = i + 1;
    }

    printf("Enter the time quantum: ");
    scanf("%d", &time_quantum);

    round_robin(processes, n, burst_time, arrival_time, time_quantum);

    return 0;
}
```