**Problem 5:** Write a program to implement the round robin scheduling algorithm having variables time quantum and find the average turnaround time, waiting time, completion time and response time for overall process. Also Print Gantt chart for it.

**Solution:**

Source Code:

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

typedef struct

{

    char process\_name[3];

    int arrival\_time;

    int burst\_time;

    int complete\_time;

    int turn\_around\_time;

    int wait\_time;

    int response\_time;

    int time\_left;

} process;

int have\_task(process arr[], int n){

    // printf("HERE");

    int have = 0,i;

    for(i=0; i<n;i++){

        if(arr[i].time\_left > 0){

            have=1;

            break;

        }

    }

    return have;

}

void print\_process\_table(process arr[],int n){

    int i;

    puts(" \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

    puts("| Process Name | Arrival Time  | Burst Time | Complete Time | Turn Around Time | Wait Time | Response Time |");

    for(i=0; i<n;i++){

        puts("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

        printf("|      %3s     |      %3d      |     %3d    |      %3d      |       %4d       |    %3d    |      %3d      |\n",

        arr[i].process\_name,arr[i].arrival\_time,arr[i].burst\_time,arr[i].complete\_time,arr[i].turn\_around\_time,arr[i].wait\_time,arr[i].response\_time);

    }

    puts("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|");

}

void get\_average(process arr[], int n){

    double tat=0,wt=0,rt=0;

    int i;

    for(i=0;i<n;i++){

        tat += (double)arr[i].turn\_around\_time;

        wt += (double)arr[i].wait\_time;

        rt += (double)arr[i].response\_time;

    }

    printf("Total time to Complete = %3d        Average Time to Complete = %.3f\n",arr[n-1].complete\_time,(double)arr[n-1].complete\_time/(double)n);

    printf("Total Turn Around Time = %.3f     Average Turn Around Time = %.3f\n",tat,tat/(double)n);

    printf("Total Waiting Time = %.3f         Average Waiting Time = %.3f\n",wt,wt/(double)n);

    printf("Total Response Time = %.3f        Average Response Time = %.3f\n",rt,rt/(double)n);

}

void gnatt(process arr[],int n,int time\_quantum){

    int i,j,time=0,total\_time=0;

    for(i=0; i<n;i++){

        arr[i].time\_left = arr[i].burst\_time;

    }

    printf("\n|");

    i=0;

    while(have\_task(arr,n)==1){

        if(arr[i].time\_left>0){

            printf("%3s  ",arr[i].process\_name);

            printf("|");

        }

        if(time\_quantum<arr[i].time\_left){

            time = time\_quantum ? time\_quantum<arr[i].time\_left : arr[i].time\_left;

        }else{

            time = arr[i].time\_left;

        }

        arr[i].time\_left-=time;

        i++;

        i%=n;

    }

    printf("\n");

    for(i=0; i<n;i++){

        arr[i].time\_left = arr[i].burst\_time;

    }

    i=0;

    while(have\_task(arr,n)==1){

        if(arr[i].time\_left>0){

            printf("%2d",total\_time);

            printf("    ",arr[i].process\_name);

        }

        if(time\_quantum<arr[i].time\_left){

            time = time\_quantum ? time\_quantum<arr[i].time\_left : arr[i].time\_left;

        }else{

            time = arr[i].time\_left;

        }

        arr[i].time\_left-=time;

        total\_time+=time;

        i++;

        i%=n;

    }

    printf("%d",total\_time);

}

void main()

{

    int n =0,i, total\_time=0, time\_quantum;

    printf("Enter the number of processes\t");

    scanf("%d",&n);

    printf("Enter the Time Quantum\t");

    scanf("%d",&time\_quantum);

    process arr[n];

    printf("Enter PROCESS\_NAME ARRIVAL\_TIME and BURST\_TIME\n");

    for(i=0; i<n;i++)

    {

        scanf("%s %d %d",arr[i].process\_name,&arr[i].arrival\_time,&arr[i].burst\_time);

        arr[i].time\_left=arr[i].burst\_time;

        // response time

        arr[i].response\_time = total\_time;

        if(arr[i].burst\_time<time\_quantum){

            total\_time += arr[i].burst\_time;

        }else{

            total\_time += time\_quantum;

        }

    }

    // complete time

    total\_time=0;

    i=0;

    while(have\_task(arr,n)==1){

        // printf("HERE IN WHILE");

        if(arr[i].time\_left > 0){

            if(time\_quantum<arr[i].time\_left){

                total\_time+=time\_quantum;

                arr[i].time\_left -= time\_quantum;

            }else{

                total\_time+=arr[i].time\_left;

                arr[i].time\_left -= arr[i].time\_left;

            }

            if(arr[i].time\_left==0){

                arr[i].complete\_time = total\_time;

                arr[i].turn\_around\_time = arr[i].complete\_time-arr[i].arrival\_time;

                arr[i].wait\_time = arr[i].turn\_around\_time-arr[i].burst\_time;

            }

        }

        i++;

        i%=n;

    }

    print\_process\_table(arr,n);

    get\_average(arr, n);

    printf("---------------------- GNATT CHART -------------------\n");

    gnatt(arr,n,time\_quantum);

}

Output:

