**Problem 8:** Write a program to implement the Highest Response Ratio Next (Non-preemptive) algorithm and find the average turnaround time, waiting time, completion time and response time for overall process.

**Answer:**

*Source Code*

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<limits.h>

#include<float.h>

typedef struct

{

char process\_name[3];

int arrival\_time;

int burst\_time;

float hrr;

int complete\_time;

int turn\_around\_time;

int wait\_time;

int response\_time;

int done;

} process;

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 i,j;

// upper row

printf(" ");

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

for(j=0;j<arr[i].burst\_time+1;j++) printf("\_\_");

printf(" ");

}

printf("\n|");

// middle row

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

for(j=0;j<arr[i].burst\_time-1;j++){

printf(" ");

}

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

for(j=0;j<arr[i].burst\_time;j++){

printf(" ");

}

printf("|");

}

printf("\n|");

// lower row

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

for(j=0;j<arr[i].burst\_time+1;j++) printf("\_\_");

printf("|");

}

printf("\n");

printf("0");

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

for(j=0; j<arr[i].burst\_time+1; j++) printf(" ");

if(arr[i].turn\_around\_time > 9) printf("\b");

printf("%d", arr[i].turn\_around\_time);

}

printf("\n");

}

int completed(process arr[], int n){

int i=0,flag=1;

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

if(arr[i].done==0){

flag=0;

break;

}

}

return flag;

}

void update\_hrr(process arr[], int n, int time){

int i=0;

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

if(arr[i].done==0){

arr[i].hrr=(float)(time-arr[i].arrival\_time+arr[i].burst\_time)/(float)arr[i].burst\_time;

}

}

}

int best\_process(process arr[], int n, int time){

int ind=-1,i=0;

float priority=FLT\_MIN;

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

if(arr[i].arrival\_time > time){

break;

}else{

if(arr[i].done==0 && arr[i].hrr>priority){

priority=arr[i].hrr;

ind=i;

}

}

}

return ind;

}

void main()

{

int n =0,i, total\_time=0,temp=0;

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

scanf("%d",&n);

process arr[n], gnt[n];

printf("Enter PROCESS\_NAME ARRIVAL\_TIME 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].done=0;

arr[i].hrr=1;

}

i=0;

while (completed(arr,n)!=1)

{

update\_hrr(arr,n,total\_time);

temp=best\_process(arr,n, total\_time); //return index of that process to execute.

if(temp==-1){

total\_time++;

}else{

arr[temp].complete\_time = total\_time+arr[temp].burst\_time;

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

arr[temp].response\_time = total\_time-arr[temp].arrival\_time;

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

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

arr[temp].done=1;

gnt[i++]=arr[temp];

}

}

print\_process\_table(arr,n);

get\_average(arr, n);

puts("---------------------- GNATT CHART -------------------");

gnatt(gnt,n);

}

*Output:*

