**Practical : 8**

**Aim: To Implement SRTN (Shortest Remaining Time Next) CPU Scheduling Algorithm.**

**Program:**

**#include <stdio.h>**

**struct process**

**{**

**int pid; int btime; int wtime; int ttime; int arrtime,comptime;**

**} p[10], temp[10];**

**int main()**

**{**

**int i,j,k,n,smallest,remain=0,time=0,end=0,c=0;**

**float awat,atur,ttur,twat,totaltime=0,throughput;**

**int cpu\_util=0;**

**printf("Enter no. of process : ");**

**scanf("%d", &n);**

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

**{**

**printf("Burst time for process P%d (in ms) : ",(i+1));**

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

**p[i].pid = i+1;**

**}**

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

**{**

**printf("enter arrival time for process P%d (in ms) : ",(i+1));**

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

**// p[i].pid = i+1;**

**}**

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

**{**

**temp[i].btime=p[i].btime;**

**}**

**p[9].btime=9999;**

**printf("\n\nGANTT Chart:\n\n");**

**for(i=1; i<10\*n +2; i++)**

**{**

**printf("-");**

**}**

**printf("\n|");**

**for(time=0;remain!=n;time++)**

**{**

**smallest=9;**

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

**{**

**if(p[i].arrtime<=time && (p[i].btime < p[smallest].btime) && p[i].btime > 0 )**

**smallest=i;**

**}**

**p[smallest].btime= p[smallest].btime - 1;**

**printf("p%d ",smallest+1);**

**printf("|");**

**if(p[smallest].btime==0)**

**{**

**remain++;**

**end=time+1;**

**p[smallest].comptime = end;**

**p[smallest].wtime = end - p[smallest].arrtime - temp[smallest].btime;**

**p[smallest].ttime = end - p[smallest].arrtime;**

**}**

**}**

**ttur = twat = 0;**

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

**{**

**ttur += p[i].ttime;**

**twat += p[i].wtime;**

**totaltime += temp[i].btime;**

**}**

**awat = (float)twat / n;**

**atur = (float)ttur / n;**

**printf("\n");**

**for(i=1; i<10\*n +2; i++)**

**{**

**printf("-");**

**}**

**printf("\n0");**

**for(i=0; i<time; i++)**

**{**

**printf(" %d",i+1);**

**}**

**printf("\n\n\n...............SRTN Scheduling...............\n\n");**

**for(i=0; i<60; i++)**

**{**

**printf("-");**

**}**

**printf("\n| Time | Process | T-Time= | W-Time= |\n");**

**printf("| taken | Completed | t(PC)-t(PS) | T.T - T(process time) | \n");**

**for(i=0; i<60; i++)**

**{**

**printf("-");**

**}**

**printf("\n");**

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

**{**

**printf("| %2d\t| P%d |\t%5d | \t%5d |",temp[i].btime,p[i].pid,p[i].ttime,p[i].wtime);**

**printf("\n+----------------------------------------------------------+\n");**

**}**

**printf("\n");**

**printf("\n\nAverage waiting time given by: total waiting time / no. of processes \nSo here: ");**

**for(i=1; i<=n; i++)**

**{**

**printf("%d",p[i].wtime);**

**if(i==n)**

**{**

**goto f;**

**}**

**printf("+");**

**f:**

**c=c+c;**

**}**

**printf("/ %d = %5.2f ms \n",n,awat);**

**printf("\n\nAverage Turn-around time given by: total turn-around time / no. of processes \nSo here: ");**

**for(i=1; i<=n; i++)**

**{**

**printf("%d",p[i].ttime);**

**if(i==n)**

**{**

**goto q;**

**}**

**printf("+");**

**q:**

**c=c+c;**

**}**

**printf("/%d = %5.2f ms \n",n,atur);**

**cpu\_util= ( totaltime/ (totaltime + 0) ) \* 100;**

**printf("\n\nCPU utilization, given by: (cpu busy time)/(cpu busytime+ idle time) \nSo here it is %5.2f / (%5.2f + 0) = %d\n\n",totaltime,totaltime, cpu\_util);**

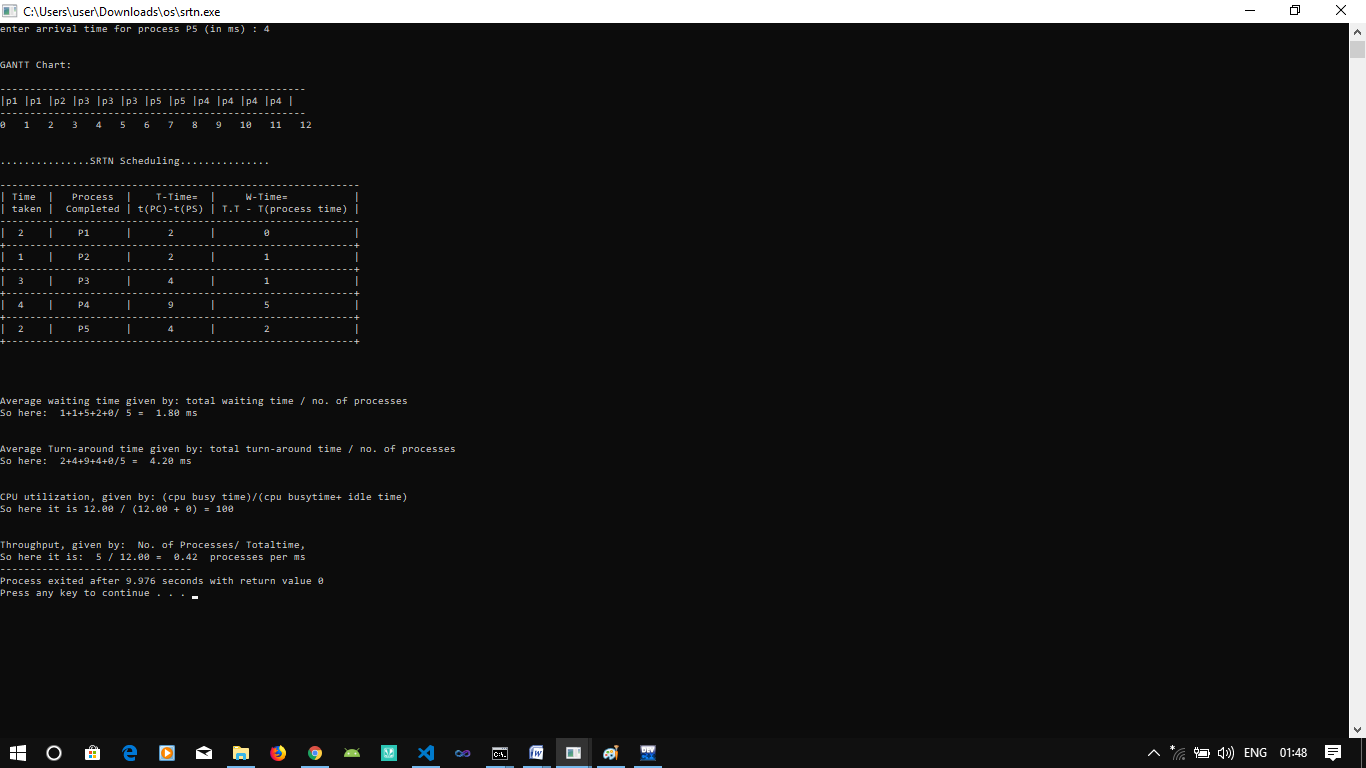
**throughput= n/ ( totaltime);**

**printf("\nThroughput, given by: No. of Processes/ Totaltime,\nSo here it is: %d / %5.2f = %5.2f processes per ms",n,totaltime,throughput);**

**return 0;**

**}**

**Output:**

****