Assignment 4: CPU scheduling algorithms

# Name – Akshat Chandrapatle Div- CS-A Roll no.- 45 PRN- 12111449

1. **FCFS**

#include <stdio.h>

void fn(int processes[], int n, int bt[]){

int wt[n], tat[n], total\_wt = 0, total\_tat = 0, fn[n];

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

fn[i] = fn[i-1]+bt[i];

}

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

tat[i] = fn[i]-processes[i];

}

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

wt[i] = tat[i] - bt[i];

}

printf("AT\t\tBT\t\tfn\t\tatAT\t\tWT\n");

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

printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i, bt[i], fn[i], tat[i], wt[i]);

}

float avg\_wt = 0.0, avg\_tat = 0.0;

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

avg\_wt += wt[i];

avg\_tat += tat[i];

}

avg\_wt /= n;

avg\_tat /= n;

printf("\nAverage WT: %0.2f", avg\_wt);

printf("\nAverage TAT: %0.2f\n", avg\_tat);

}

int main(){

int processes[] = {0,2,3,4,5};

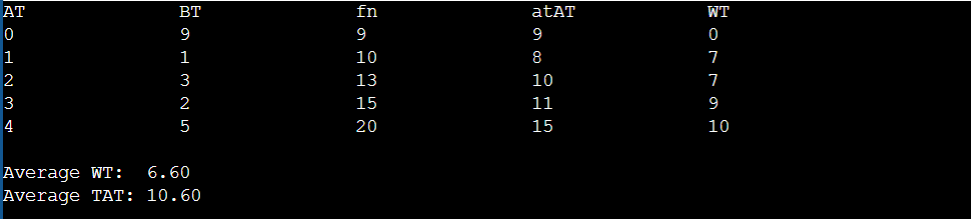
int n = sizeof(processes) / sizeof(processes[0]);

int bt[] = {9,1,3,2,5};

fn(processes, n, bt);

}

Output:



1. **Priority (preemptive )**

Code:

#include<stdio.h>

#define MAX 9999;

struct proc{

int no,at,bt,ct,wt,tat,pri,status;

};

struct proc read(int i){

struct proc p;

printf("\nProcess No: %d\n",i);

p.no=i;

printf("Enter Arrival Time: ");

scanf("%d",&p.at);

printf("Enter Burst Time: ");

scanf("%d",&p.bt);

printf("Enter Priority: ");

scanf("%d",&p.pri);

p.status=0;

return p;

}

int main(){

int n,s,ct=0,remaining;

struct proc p[10],temp;

float avgtat=0,avgwt=0;

printf("Enter Number of Processes: ");

scanf("%d",&n);

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

p[i]=read(i+1);

for(int i=0;i<n-1;i++)

for(int j=0;j<n-i-1;j++)

if(p[j].at>p[j+1].at)

{

temp=p[j];

p[j]=p[j+1];

p[j+1]=temp;

}

p[9].pri=MAX;

remaining=n;

printf("\nProcessNo\tAT\tBT\tPri\tCT\tTAT\tWT\tRT\n");

for(ct=p[0].at;remaining!=0;)

{

s=9;

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

if(p[i].at<=ct && p[i].status!=1 && p[i].pri<p[s].pri)

s=i;

p[s].ct=ct=ct+p[s].bt;

p[s].tat=p[s].ct-p[s].at;

avgtat+=p[s].tat;

p[s].wt=p[s].tat-p[s].bt;

avgwt+=p[s].wt;

p[s].status=1;

remaining--;

printf("P%d\t\t%d\t%d\t%d\t%d\t%d\t%d\t%d\n",p[s].no,p[s].at,p[s].bt,p[s].pri,p[s].ct,p[s].tat,p[s].wt,p[s].wt);

}

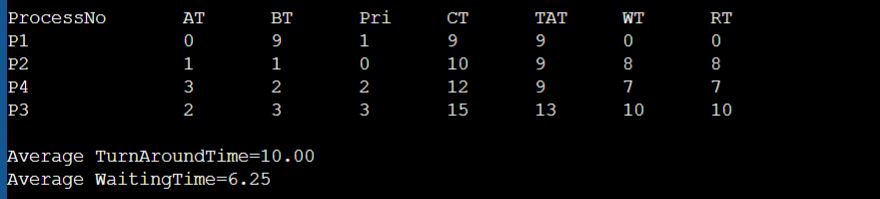
avgtat/=n,avgwt/=n;

printf("\nAverage TurnAroundTime=%0.2f\nAverage WaitingTime=%0.2f\n",avgtat,avgwt);

return 0;

}

Output:



1. **Priority (non prepemptive)**

Code:

#include<stdio.h>

#define MAX 9999;

struct proc{

int no,at,bt,rt,ct,wt,tat,pri,temp;

};

struct proc read(int i){

struct proc p;

printf("\nProcess No: %d\n",i);

p.no=i;

printf("Enter Arrival Time: ");

scanf("%d",&p.at);

printf("Enter Burst Time: ");

scanf("%d",&p.bt);

p.rt=p.bt;

printf("Enter Priority: ");

scanf("%d",&p.pri);

p.temp=p.pri;

return p;

}

int main(){

int i,n,c,remaining,min\_val,min\_index;

struct proc p[10],temp;

float avgtat=0,avgwt=0;

printf("Enter Number of Processes: ");

scanf("%d",&n);

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

p[i]=read(i+1);

remaining=n;

for(int i=0;i<n-1;i++)

for(int j=0;j<n-i-1;j++)

if(p[j].at>p[j+1].at){

temp=p[j];

p[j]=p[j+1];

p[j+1]=temp;

}

min\_val=p[0].temp,min\_index=0;

for(int j=0;j<n&&p[j].at<=p[0].at;j++)

if(p[j].temp<min\_val)

min\_val=p[j].temp,min\_index=j;

i=min\_index;

c=p[i].ct=p[i].at+1;

p[i].rt--;

if(p[i].rt==0){

p[i].temp=MAX;

remaining--;

}

while(remaining>0){

min\_val=p[0].temp,min\_index=0;

for(int j=0;j<n&&p[j].at<=c;j++)

if(p[j].temp<min\_val)

min\_val=p[j].temp,min\_index=j;

i=min\_index;

p[i].ct=c=c+1;

p[i].rt--;

if(p[i].rt==0)

{

p[i].temp=MAX;

remaining--;

}

}

printf("\nProcessNo\tAT\tBT\tPri\tCT\tTAT\tWT\n");

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

p[i].tat=p[i].ct-p[i].at;

avgtat+=p[i].tat;

p[i].wt=p[i].tat-p[i].bt;

avgwt+=p[i].wt;

printf("P%d\t\t%d\t%d\t%d\t%d\t%d\t%d\n",p[i].no,p[i].at,p[i].bt,p[i].pri,p[i].ct,p[i].tat,p[i].wt);

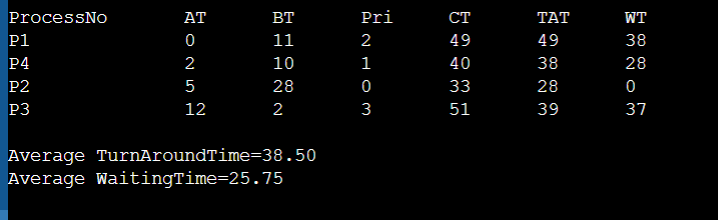
}

avgtat/=n,avgwt/=n;

printf("\nAverage TurnAroundTime=%0.2f\nAverage WaitingTime=%0.2f\n",avgtat,avgwt);

return 0;

}



1. **SJF (preemptive)**

Code:

#include <stdio.h>

int main()

{

int at[10], bt[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, tt = 0, end;

float average\_waiting\_time, average\_tt;

printf("\nEnter the Total Number of Processes: ");

scanf("%d", &limit);

printf("\nEnter Details of %d Processes\n", limit);

for(i = 0; i < limit; i++)

{

printf("\nEnter Arrival Time: ");

scanf("%d", &at[i]);

printf("Enter Burst Time: ");

scanf("%d", &bt[i]);

temp[i] = bt[i];

}

bt[9] = 9999;

for(time = 0; count != limit; time++)

{

smallest = 9;

for(i = 0; i < limit; i++)

{

if(at[i] <= time && bt[i] < bt[smallest] && bt[i] > 0)

{

smallest = i;

}

}

bt[smallest]--;

if(bt[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - at[smallest] - temp[smallest];

tt = tt + end - at[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

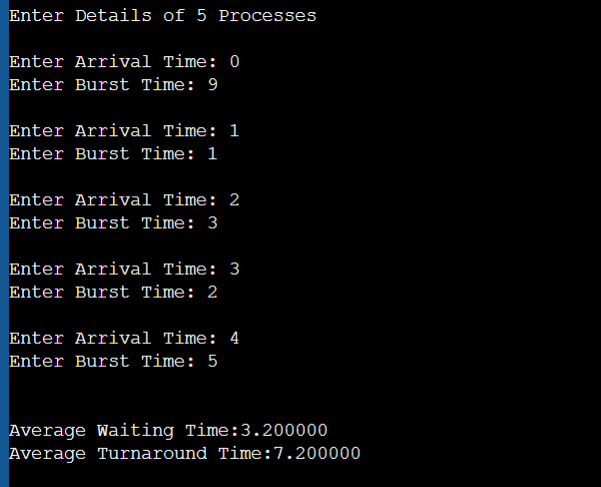
average\_tt = tt / limit;

printf("\n\nAverage Waiting Time:%lf\n", average\_waiting\_time);

printf("Average Turnaround Time:%lf\n", average\_tt);

return 0;

}



1. SJF (non prepemptive)

Code:

#include <stdio.h>

int main(){

int at[] = {0, 1, 3, 3};

int bt[] = {1, 9, 1, 9};

int n = sizeof(at)/ sizeof(at[0]);

int ft[n], tat[n], wt[n];

int total\_wt = 0, total\_tat = 0;

int i, j, min, temp;

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

min = i;

for (j = i + 1; j < n; j++){

if (at[j] < at[min]){

min = j;

}

}

temp = at[i];

at[i] = at[min];

at[min] = temp;

temp = bt[i];

bt[i] = bt[min];

bt[min] = temp;

}

ft[0] = at[0] + bt[0];

tat[0] = ft[0] - at[0];

wt[0] = tat[0] - bt[0];

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

ft[i] = ft[i - 1] + bt[i];

tat[i] = ft[i] - at[i];

wt[i] = tat[i] - bt[i];

}

printf("AT\t BT\t FT\t TAT\t WT");

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

printf("\n%d\t %d\t %d\t %d\t %d", at[i], bt[i], ft[i], tat[i], wt[i]);

total\_wt += wt[i];

total\_tat += tat[i];

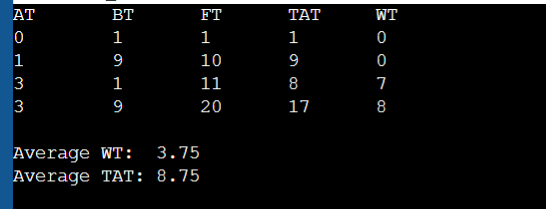
}

printf("\n\nAverage WT: %0.2f", (total\_wt \* 1.0 / n));

printf("\nAverage TAT: %0.2f\n", (total\_tat \* 1.0 / n));

return 0;

}



1. **Round robin**

Code:

#include <stdio.h>

int main()

{

int i, n, time, remain, flag = 0, tq = 1;

int wt = 0, tt = 0, ft[10], rt[10];

int at[] = {0,1,3,3};

int bt[] = {1,9,1,9};

n = sizeof at / sizeof at[0];

remain = n;

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

rt[i] = bt[i];

}

printf("AT\t BT\t FT\t TAT\t WT");

for (time = 0, i = 0; remain != 0;){

if (rt[i] <= tq && rt[i] > 0){

time += rt[i];

rt[i] = 0;

flag = 1;

}else if (rt[i] > 0){

rt[i] -= tq;

time += tq;

}

if (rt[i] == 0 && flag == 1){

remain--;

ft[i] = time;

printf("\n%d\t %d\t %d\t %d\t %d", at[i],bt[i], ft[i], ft[i] - at[i], ft[i] - at[i] - bt[i]);

wt += ft[i] - at[i] - bt[i];

tt += ft[i] - at[i];

flag = 0;

}

if (i == n - 1){

i = 0;

} else if (at[i + 1] <= time){

i++;

} else{ i = 0; }

}

printf("\n\nAverage WT: %0.2f", (wt \* 1.0 / n));

printf("\nAverage TAT: %0.2f\n", (tt \* 1.0 / n));

return 0;

}

Output:

