OS LAB- 1BM21CS238

WEEK 2

First come first serve algorithm:-

Code:-

#include<stdio.h>

typedef struct

{

int pID,aT,bT,sT,cT,taT,wT;

} Process;

void calculateTimes(Process p[], int n)

{

int currT = 0;

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

{

p[i].sT = currT;

p[i].cT = currT + p[i].bT;

p[i].taT = p[i].cT - p[i].aT;

p[i].wT = p[i].taT - p[i].bT;

currT = p[i].cT;

}

}

void displayp(Process p[], int n)

{

printf("Process\tArrival Time\tBurst Time\tStart Time\tCompletion Time\tTurnaround Time\tWaiting Time\n");

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

{

printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].pID, p[i].aT,

p[i].bT, p[i].sT, p[i].cT,

p[i].taT, p[i].wT);

}

}

void averageWaitingTime(Process p[], int n){

printf("The average waiting time of all %d processes are :\n",n);

float sum=0.0;

int k;

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

sum+=p[k].wT;

}

float avg = (sum/n);

printf("%f",avg);

}

int main() {

int n;

printf("Enter the number of processes: ");

scanf("%d", &n);

Process p[n];

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

printf("Enter the arrival time and burst time for process %d: ", i + 1);

scanf("%d %d", &p[i].aT, &p[i].bT);

p[i].pID = i + 1;

}

calculateTimes(p, n);

displayp(p, 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) {

Process temp = p[j];

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

p[j + 1] = temp;

}

}

}

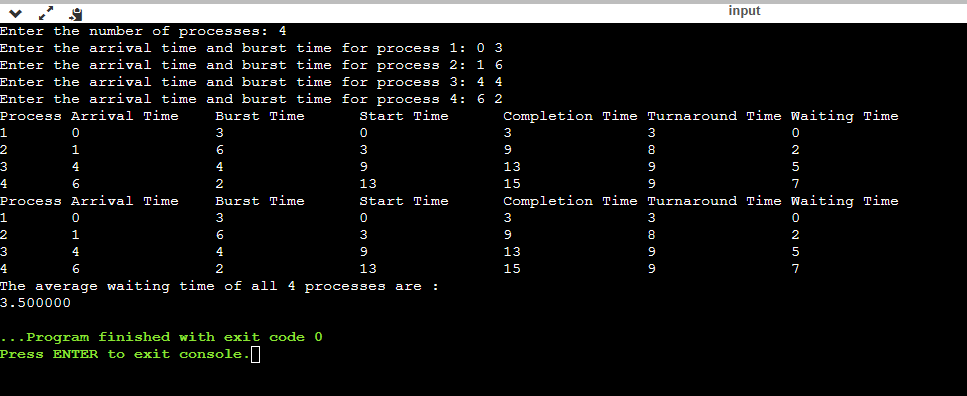
calculateTimes(p, n);

displayp(p, n);

averageWaitingTime(p, n);

return 0;

}

OUTPUT:- 

Shortest Job First(SJF) :-

Code:-

#include<stdio.h>

typedef struct

{

int pID,aT,bT,sT,cT,taT,wT;

} Process;

void calculateTimes(Process p[], int n)

{

int i,j,t, s,f;

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

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

s = (p[j].bT)+(p[j].aT);

f = (p[j+1].bT)+(p[j+1].aT);

if(s > f){

t=f;

f= s;

s = t;

}

}

}

int currT = 0;

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

{

p[i].sT = currT;

p[i].cT = currT + p[i].bT;

p[i].taT = p[i].cT - p[i].aT;

p[i].wT = p[i].taT - p[i].bT;

currT = p[i].cT;

}

}

void displayp(Process p[], int n)

{

printf("Process\tArrival Time\tBurst Time\tStart Time\tCompletion Time\tTurnaround Time\tWaiting Time\n");

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

{

printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].pID, p[i].aT,

p[i].bT, p[i].sT, p[i].cT,

p[i].taT, p[i].wT);

}

}

void averageWaitingTime(Process p[], int n){

printf("The average waiting time of all %d processes are :\n",n);

float sum=0.0;

int k;

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

sum+=p[k].wT;

}

float avg = (sum/n);

printf("%f",avg);

}

int main() {

int n;

printf("Enter the number of processes: ");

scanf("%d", &n);

Process p[n];

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

printf("Enter the arrival time and burst time for process %d: ", i + 1);

scanf("%d %d", &p[i].aT, &p[i].bT);

p[i].pID = i + 1;

}

calculateTimes(p, n);

displayp(p, 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) {

Process temp = p[j];

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

p[j + 1] = temp;

}

}

}

calculateTimes(p, n);

displayp(p, n);

averageWaitingTime(p, n);

return 0;

}

OUTPUT:-

