**Bahria University,**

**Karachi Campus**



**LAB EXPERIMENT NO.**

**07**

**LIST OF TASKS**

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| **01** | **The FCFS source code given above assumes that the arrival times for processes are provided in ascending order of time. Modify the code in C language for random process arrival times.** |
| **02** | **Modify the FCFS code of example#02 to be able to calculate the mean of Normalized Turnaround Time. Also the modified code should display the Normalized Turnaround Time of each process.** |
| 03 | Implement the Shortest Job First algorithm with the help of 4 non-preemptive processes. |

**Submitted On:**

\_\_\_\_\_\_\_\_\_\_\_\_

**(Date: DD/MM/YY)**

**Task No. 1:** The FCFS source code given above assumes that the arrival times for processes are provided in ascending order of time. Modify the code in C language for random process arrival times.

**Solution:**

#include<stdio.h>

#include<stdlib.h>

void Sort(int at[],int st[],int pn[],int n){

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

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

if (at[i] > at[i + 1]){ int temp = at[i];

at[i] = at[i + 1]; at[i + 1] = temp;

int temp2 = st[i]; st[i] = st[i + 1];

st[i + 1] = temp2; int temp3 = pn[i];

pn[i] = pn[i + 1]; pn[i + 1] = temp3;}} }}

void main (){

int at[20], st[20], wt[20], ft[20], tat[20], i, n,pn[20];

float wt\_avg, tat\_avg;

printf("\nEnter the number of processes: ");

scanf("%d", &n);

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

at[i]=rand(); pn[i]=i+1;

printf("\nArrival time for process%d: %d", pn[i],at[i]);

printf("\nEnter Burst/Service time for process%d: ", pn[i]);

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

Sort(at,st,pn,n); wt[0] = wt\_avg = 0;

tat[0] = tat\_avg = st[0]; ft[0] = st[0];

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

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

wt[i] = wt[i-1] + st[i-1]; ft[i] = wt[i] + st[i];

tat[i] = ft[i] - at[i]; wt\_avg = wt\_avg + wt[i];

tat\_avg = tat\_avg + tat[i]; } }

printf("\n PROCESS \t Arrival Time \t SERVICE TIME \t WAITING TIME \t FINISH TIME \t TURNAROUND TIME\n");

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

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

printf("\nAverage waiting time:%f ", wt\_avg/n);

printf("\nAverage turnaround time:%f \n", tat\_avg/n); }

**Output:**

A screenshot of a computer program

Description automatically generated with medium confidence

**Task No. 2:** Modify the FCFS code of example#02 to be able to calculate the mean of Normalized Turnaround Time. Also the modified code should display the Normalized Turnaround Time of each process.

**Solution:**

#include<stdio.h>

void main (void){

int at[20], st[20], wt[20], ft[20], tat[20], i, n;

float wt\_avg, tat\_avg, ntat\_avg, ntat[20];

printf("\nEnter the number of processes: ");

scanf("%d", &n);

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

printf("\nEnter the Arrival time for process%d: ", i);

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

printf("\nEnter Burst/Service time for process%d: ", i);

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

ntat\_avg=0; wt[0] = wt\_avg = 0;

tat[0] = tat\_avg = st[0]; ft[0] = st[0];

ntat[0]=tat[0]/st[0];

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

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

wt[i] = wt[i-1] + st[i-1]; ft[i] = wt[i] + st[i];

tat[i] = ft[i] - at[i]; ntat[i]=tat[i]/st[i];

wt\_avg = wt\_avg + wt[i]; tat\_avg = tat\_avg + tat[i];

ntat\_avg = ntat\_avg + ntat[i]; } }

printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t FINISH TIME \t TURNAROUND TIME \t Normalize Turnaround Time\n");

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

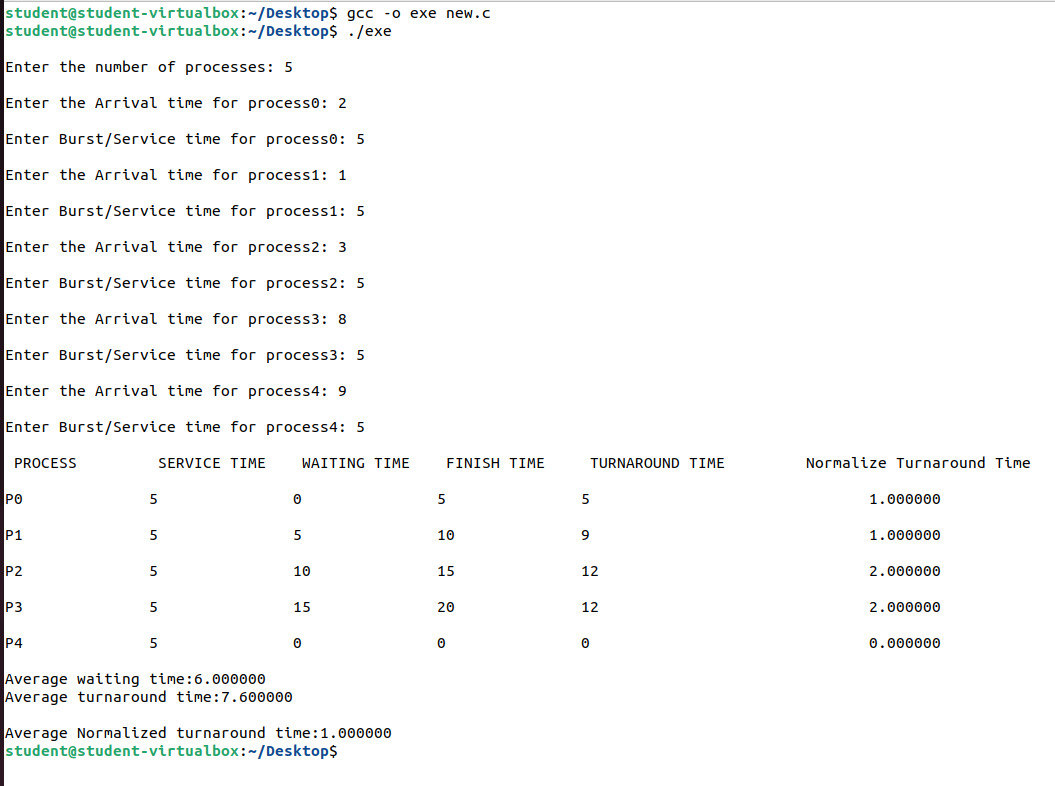
printf("\nP%d \t\t%d \t\t%d \t\t%d \t\t%d \t\t\t\t%f\n", i, st[i], wt[i], ft[i], tat[i], ntat[i]); }

printf("\nAverage waiting time:%f ", wt\_avg/n);

printf("\nAverage turnaround time:%f \n", tat\_avg/n);

printf("\nAverage Normalized turnaround time:%f \n", ntat\_avg/n); }

**Output:**



**Task No. 3:** **Implement the Shortest Job First algorithm with the help of 4 non-preemptive processes.**

**Solution:**

#include<stdio.h>

void Sort(int st[],int pn[],int n){

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

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

if (st[i] > st[i + 1]) {

int temp2 = st[i]; st[i] = st[i + 1];

st[i + 1] = temp2; int temp3 = pn[i];

pn[i] = pn[i + 1]; pn[i + 1] = temp3; } } } }

void main (void) {

int pn[20], at[20],st[20], wt[20], ft[20], tat[20], i, n;

float wt\_avg, tat\_avg;

printf("\nEnter the number of processes: ");

scanf("%d", &n);

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

pn[i]=i+1; at[i]=0;

printf("\nEnter Burst/Service time for process%d: ", pn[i]);

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

Sort(st,pn,n); wt[0] = wt\_avg = 0;

tat[0] = tat\_avg = st[0]; ft[0] = st[0];

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

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

wt[i] = wt[i-1] + st[i-1]; ft[i] = wt[i] + st[i];

tat[i] = ft[i] - at[i]; wt\_avg = wt\_avg + wt[i];

tat\_avg = tat\_avg + tat[i]; } }

printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t FINISH TIME \t TURNAROUND TIME\n");

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

printf("\nP%d \t\t%d \t\t%d \t\t%d \t\t%d \n", pn[i], st[i], wt[i], ft[i], tat[i]); }

printf("\nAverage waiting time:%f ", wt\_avg/n);

printf("\nAverage turnaround time:%f \n", tat\_avg/n); }

**Output:**

