SRTF

#include <stdio.h>

int arr[10], bur[10], rt[10], n, pr[10], t = 0, ft[10], wt[10], tat[10], pos, total = 0, ttat = 0, twt = 0;

float avgtat, avgwt;

int main() {

int comp = 0, i;

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

scanf("%d", &n);

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

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

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

pr[i] = i + 1;

}

for (i = 0; i < n; i++) rt[i] = bur[i];

while (comp < n) {

pos = -1;

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

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

if (pos == -1 || rt[i] < rt[pos]) {

pos = i;

}

}

}

if (pos == -1) {

t++;

continue;

}

rt[pos]--;

t++;

if (rt[pos] == 0) {

comp++;

ft[pos] = t;

tat[pos] = ft[pos] - arr[pos];

wt[pos] = tat[pos] - bur[pos];

}

}

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

ttat += tat[i];

twt += wt[i];

}

avgtat = (float)ttat / n;

avgwt = (float)twt / n;

printf("\nProcess\tBT\tAT\tTAT\tWT\n");

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

printf("%d\t%d\t%d\t%d\t%d\n", pr[i], bur[i], arr[i], tat[i], wt[i]);

}

printf("Avg TAT = %.2f\nAvg WT = %.2f\n", avgtat, avgwt);

return 0;

}

RR

#include <stdio.h>

#include <stdlib.h>

struct J {

int bt, tat, wt, at, ft;

} job[100];

void scheduler(struct J job[], int n, int q) {

int burst[100], t = 0, done = 0, curr = -1, diff = q, i = 0;

float tat\_sum = 0, wt\_sum = 0;

for (i = 0; i < n; i++) burst[i] = job[i].bt;

while (done < n) {

while (1) {

curr = (curr + 1) % n;

if (job[curr].bt != 0) break;

}

diff = (q <= job[curr].bt) ? q : job[curr].bt;

job[curr].bt -= diff;

t += diff;

if (job[curr].bt == 0) {

done++;

job[curr].ft = t;

}

}

printf("RR Scheduling Details are \n");

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

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

job[i].wt = job[i].tat - burst[i];

tat\_sum += job[i].tat;

wt\_sum += job[i].wt;

}

printf("Job\tBT\tAT\tTAT\tWT\n");

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

printf("%d\t%d\t%d\t%d\t%d\n", i + 1, burst[i], job[i].at, job[i].tat, job[i].wt);

printf("Avg TAT=%f\nAvg WT=%f\n", tat\_sum / n, wt\_sum / n);

}

void main() {

int n, q, i;

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

scanf("%d", &n);

printf("Enter the arrival time and burst time\n");

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

printf("Job%d: ", i + 1);

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

}

printf("Enter time quantum: ");

scanf("%d", &q);

scheduler(job, n, q);

}

BANK

#include <stdio.h>

void displayMatrix(int matrix[][10], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

printf("%d ", matrix[i][j]);

}

printf("\n");

}

}

int main() {

int Max[10][10], alloc[10][10], need[10][10], avail[10], completed[10], safeSequence[10];

int p, r, i, j, process, count = 0;

do {

printf("Enter the number of processes (max 10): ");

scanf("%d", &p);

} while (p <= 0 || p > 10);

do {

printf("Enter the number of resources (max 10): ");

scanf("%d", &r);

} while (r <= 0 || r > 10);

for (i = 0; i < p; i++) completed[i] = 0;

printf("Enter the Max Matrix for each process:\n");

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

printf("For process %d: ", i + 1);

for (j = 0; j < r; j++) scanf("%d", &Max[i][j]);

}

printf("Enter the allocation for each process:\n");

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

printf("For process %d: ", i + 1);

for (j = 0; j < r; j++) scanf("%d", &alloc[i][j]);

}

printf("Enter the Available Resources:\n");

for (i = 0; i < r; i++) scanf("%d", &avail[i]);

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

for (j = 0; j < r; j++)

need[i][j] = Max[i][j] - alloc[i][j];

printf("\nMax Matrix:\n");

displayMatrix(Max, p, r);

printf("\nAllocation Matrix:\n");

displayMatrix(alloc, p, r);

printf("\nNeed Matrix:\n");

displayMatrix(need, p, r);

do {

process = -1;

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

if (completed[i] == 0) {

process = i;

for (j = 0; j < r; j++) {

if (avail[j] < need[i][j]) {

process = -1;

break;

}

}

}

if (process != -1) break;

}

if (process != -1) {

printf("\nProcess %d runs to completion!", process + 1);

for (j = 0; j < r; j++) {

avail[j] += alloc[process][j];

alloc[process][j] = 0;

Max[process][j] = 0;

}

completed[process] = 1;

safeSequence[count] = process + 1;

count++;

}

} while (count != p && process != -1);

if (count == p) {

printf("\nThe system is in a safe state!!\n");

printf("Safe Sequence : < ");

for (i = 0; i < p; i++) printf("%d ", safeSequence[i]);

printf(">\n");

} else {

printf("\nThe system is in an unsafe state!!\n");

}

return 0;

}

FIFO

#include <stdio.h>

int n, pg[30], fr[10];

void fifo();

int main() {

int i;

printf("Enter total number of pages: ");

scanf("%d", &n);

printf("Enter page sequence:\n");

for (i = 0; i < n; i++) scanf("%d", &pg[i]);

fifo();

return 0;

}

void fifo() {

int i, f = 0, r = 0, s = 0, count = 0, flag = 0, num, psize;

printf("Enter the size of page frame: ");

scanf("%d", &psize);

for (i = 0; i < psize; i++) fr[i] = -1;

while (s < n) {

flag = 0;

num = pg[s];

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

if (num == fr[i]) {

s++;

flag = 1;

break;

}

}

if (flag == 0) {

if (r < psize) {

fr[r] = pg[s];

r++;

s++;

count++;

} else {

if (f < psize) {

fr[f] = pg[s];

s++;

f++;

count++;

}

f = 0;

}

}

}

printf("\nPage Frame: ");

for (i = 0; i < psize; i++) printf("%d ", fr[i]);

printf("\nPage Faults: %d\n", count);

}