Cho n tiến trình với thời gian xử lý (burst times) của chúng, nhiệm vụ là tìm thời gian chờ trung bình và thời gian quay vòng trung bình bằng thuật toán điều phối FCFS.

Đến trước xử lý trước (FIFO), còn được gọi là đến trước phục vụ trước (FCFS), là thuật toán điều phối thời gian đơn giản nhất. FIFO chỉ cần xếp hàng các tiến trình theo thứ tự mà chúng đến trong hàng đợi sẵn sàng.

Trong đó, tiến trình đến trước sẽ được thực thi trước và tiến trình tiếp theo chỉ bắt đầu sau khi tiến trình trước được thực thi đầy đủ.

Completion Time (Thời gian hoàn thành): Thời gian mà tiến trình hoàn thành việc thực hiện.

Turn Around Time (Thời gian quay vòng): Thời gian Chênh lệch giữa thời gian hoàn thành và thời gian đến.

* Turn Around Time = Completion Time – Thời gian đến (Arrival Time)

Thời gian chờ (Waiting Time): Thời gian Chênh lệch giữa thời gian quay vòng và thời gian xử lý (burst times)

* Thời gian chờ = Turn Around Time - burst times

Ví dụ:

|  |  |  |
| --- | --- | --- |
| Process | Burst times | Arrival Time |
| P1 | 23 | 3 |
| P2 | 4 | 2 |
| P3 | 5 | 1 |

Yêu cầu thực hiện viết 1 chương trình bằng ngôn ngữ C mô phỏng thuật toán FCFS?

1- Nhập các tiến trình cùng với burst time (bt).

2- Tìm thời gian chờ (wt) cho tất cả các tiến trình.

wt[i] = bt[i-1] + wt[i-1] .

3- Tìm thời gian quay vòng =waiting\_time +burst\_time cho tất cả các tiến trình.

4- Tìm thời gian chờ trung bình = tổng\_thời\_gian chờ / số\_tiến\_trình.

5- Tương tự, tìm thời gian quay vòng trung bình = tổng\_ turn\_around\_time / số\_tiến\_trình

#include <stdio.h>

#define MAX 10

struct Process {

int burst\_time;

int arrival\_time;

int waiting\_time;

int turnaround\_time;

};

void findWaitingTime(struct Process p[], int n) {

p[0].waiting\_time = 0;

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

p[i].waiting\_time = p[i - 1].waiting\_time + p[i - 1].burst\_time;

}

}

void findTurnaroundTime(struct Process p[], int n) {

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

p[i].turnaround\_time = p[i].waiting\_time + p[i].burst\_time;

}

}

void findAverageTime(struct Process p[], int n) {

int total\_waiting\_time = 0;

int total\_turnaround\_time = 0;

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

total\_waiting\_time += p[i].waiting\_time;

total\_turnaround\_time += p[i].turnaround\_time;

}

float avg\_waiting\_time = (float) total\_waiting\_time / n;

float avg\_turnaround\_time = (float) total\_turnaround\_time / n;

printf("\nThoi gian cho trung binh: %.2f", avg\_waiting\_time);

printf("\nThoi gian quay vong trung binh: %.2f", avg\_turnaround\_time);

}

int main() {

int n;

printf("Nhap so long tien trinh: ");

scanf("%d", &n);

struct Process p[n];

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

printf("\nNhap thoi gian xu ly và thoi gian den cho tien trinh P%d: ", i + 1);

scanf("%d %d", &p[i].burst\_time, &p[i].arrival\_time);

}

findWaitingTime(p, n);

findTurnaroundTime(p, n);

printf("\nTien Trinh\tThoi gian xu ly\tThoi gian den\tThoi gian cho\tThoi gian quay vong\n");

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

printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i + 1, p[i].burst\_time, p[i].arrival\_time, p[i].waiting\_time, p[i].turnaround\_time);

}

findAverageTime(p, n);

return 0;

}