



**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**

**🙢 🕮 🙠**

**-Logo

Description automatically generated**

**THỰC HÀNH HỆ ĐIỀU HÀNH**

**Bài thực hành số 4**

**Họ và tên: Lê Anh Thư**

**MSSV: 20521985**

**Giảng viên: Nguyễn Thanh Nam**

**4.3.5 Câu hỏi chuẩn bị**

**1. Vẽ sơ đồ giải thuật của các giải thuật lập lịch tiến trình:**

**Bài làm**

❖ FCFS (First Come First Served)

Diagram

Description automatically generated

❖ RR (Round Robin)

Diagram

Description automatically generated

❖ SJF (Shortest Job First)

Diagram

Description automatically generated

❖ SRT (Shortest Remain Time)

Diagram

Description automatically generated

**2. Giải thích các thuật ngữ sau:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Thuật ngữ** | **Mô tả** |
| **1** | **Arrival time** | Thời gian project đến yêu cầu hệ điều hành |
| **2** | **Brust time** | Thời gian cần thiết của CPU để hoàn thành quá trình thực hiện của nó. |
| **3** | **Quantum time (time slice)** | Là lượng thời gian CPU được chỉ định để thực thi , độc lập với thời gian hoàn thành thực tế. Đối với thuật giải RR, mỗi tiến trình trước khi bắt đầu được đưa vào CPU xử lý, sẽ được cấp phát cho một đơn vị thời gian chiếm dụng CPU nhất định. Như vậy, khi một tiến trình sử dụng hết thời gian quantum mà nó được cấp phát, thì dù vẫn còn phải xử lý tiếp, phần dư của nó cũng sẽ được chuyển về phía sau trong danh sách hàng đợi. |
| **4** | **Response time** | Thời gian hệ điều hành đáp ứng. Là thời gian từ lúc có yêu cầu của người dùng (user request) đến khi có đáp ứng đầu tiên |
| **5** | **Waiting time** | Thời gian chờ. Là thời gian một process ở trong hàng đợi ready |
| **6** | **Turnaround time** | Là khoảng thời gian từ thời điểm yêu cầu của một quá trình đến thời điểm kết thúc của quá trình. Nó cũng có thể được coi như là tổng các khoảng thời gian chờ đợi để được vào bộ nhớ hoặc hàng đợi sẵn sàng cùng với thời gian thực hiện trên CPU và thực thi đầu vào / đầu ra. |
| **7** | **Average waiting time** | Thời gian chờ trung bình của các project trong tiến trình |
| **8** | **Average turnaround time** | Thời gian quay vòng trung bình của các project trong tiến trình |

**4.4 Hướng dẫn thực hành**

Soạn thảo và biên dịch giải thuật FCFS bên dưới:

Thực thi chương trình trên với đầu vào như sau:

- Số lượng tiến trình: 4

- Tiến trình đầu tiên (tên tiến trình, thời gian đến, thời gian xử lý): 0 0 2

- Tiến trình thứ hai: 1 1 3

- Tiến trình thứ ba: 2 2 4

Kiểm tra kết quả của chương trình có giống như bên dưới hay không?

A picture containing chart

Description automatically generated

Tự kiểm tra kết quả của chương trình và gỡ lỗi nếu cần thiết. Sau đó bổ sung code để tính average waiting time và average turnaround.

**Bài làm**

* **Kết quả sau khi thực thi chương trình**

Text

Description automatically generated

Kết luận: Kết quả của chương trình giống với kết quả của đề bài

* **Bổ sung code để tính average waiting time và average turnaround.**

**Source code bổ sung:**

float awt = 0;

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

awt = awt + wt[i]; }

awt = awt/n;

float att = 0;

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

att = att + tat[i]; }

att = att/n;

printf("\nAverage waiting time: %f", awt);

printf("\nAverage turnaround time: %f", att);

**Source code hoàn thiện:**

/\*######################################

# University of Information Technology #

# IT007 Operating System #

10

# <Your name>, <your Student ID> #

# File: fcfs.c #

######################################\*/

#include<stdio.h>

void main(){

    int pn[10];

    int arr[10], bur[10], star[10], finish[10], tat[10], wt[10], i, n;

    int totwt=0, tottat=0;

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

    scanf("%d",&n);

    for(i=0;i<n;i++) {

        printf("Enter the Process Name, Arrival Time & Burst Time:");

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

    }

    for(i=0;i<n;i++) {

        if(i==0) {

            star[i]=arr[i];

            wt[i]=star[i]-arr[i];

            finish[i]=star[i]+bur[i];

            tat[i]=finish[i]-arr[i];

        }

        else{

            star[i]=finish[i-1];

            wt[i]=star[i]-arr[i];

            finish[i]=star[i]+bur[i];

            tat[i]=finish[i]-arr[i];

        }

    }

    printf("\nPName Arrtime Burtime Start TAT Finish");

    for(i=0;i<n;i++) {

        printf("\n%d\t%6d\t%6d\t%6d\t%6d\t%6d",pn[i],arr[i],bur[i],star[i],tat[i],finish[i]);

        totwt+=wt[i];

        tottat+=tat[i];

    }

    float awt = 0;

    for (i=0; i < n; i++){

        awt = awt + wt[i];

    }

    awt = awt/n;

    float att = 0;

    for (i=0; i<n; i++){

        att = att + tat[i];

    }

    att = att/n;

    printf("\nAverage waiting time: %f", awt);

    printf("\nAverage turnaround time: %f", att);

    return;

}

**Kết quả:**

A picture containing text

Description automatically generated

**4.5 Bài tập ôn tập**

**1. Viết chương trình mô phỏng giải thuật SJF với các yêu cầu sau:**

❖ Nhập số lượng process

❖ Nhập process name, arrival time, burst time

❖ In ra Process name, response time, waiting time, turnaround time, average waiting time, average turnaround time

**Source code :**

#include<stdio.h>

int main()

{

    int time, pn[10], bt[10], at[10], sum\_bt=0, smallest, n, i;

    int sum\_turnaround=0,sum\_wait=0;

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

    scanf("%d",&n);

    for(i=0;i<n;i++)

    {

        printf("Enter the Process Name, Arrival Time & Burst Time: ");

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

        sum\_bt+=bt[i];

    }

    bt[9]=9999;

    printf("\nPName\t|Response Time\t|Waiting Time\t|Turnaround Time\n");

    for(time=0;time<sum\_bt;) {

        smallest=9;

        for(i=0;i<n;i++) {

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

                smallest=i;

        }

        if(smallest==9) {

            time++;

            continue;

        }

        printf("%d\t|\t%d\t|\t%d\t|\t%d\n", pn[smallest], time-at[smallest], time-at[smallest], time+bt[smallest]-at[smallest]);

        sum\_turnaround+=time+bt[smallest]-at[smallest];

        sum\_wait+=time-at[smallest];

        time+=bt[smallest];

        bt[smallest]=0;

    }

    printf("\n\nAverage waiting time = %f",sum\_wait\*1.0/n);

    printf("\n\nAverage turnaround time = %f",sum\_turnaround\*1.0/n);

    return 0;

}

**Kết quả:**

Text

Description automatically generated

**2. Viết chương trình mô phỏng giải thuật SRT với các yêu cầu sau:**

❖ Nhập số lượng process

❖ Nhập process name, arrival time, burst time 13

❖ In ra Process name, response time, waiting time, turnaround time, average waiting time, average turnaround time

**Source code :**

#include <stdio.h>

int main()

{

    int arrival\_time[10], burst\_time[10], temp[10], rt[10], pn[10], wt, tat;

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

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

    float average\_waiting\_time, average\_turnaround\_time;

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

    scanf("%d", &limit);

    for(i = 0; i < limit; i++) {

        printf("\nEnter the Process Name, Arrival Time & Burst Time: ");

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

        rt[i] = -1;

        temp[i] = burst\_time[i];

    }

    temp[9] = 9999;

    printf("PName\t|Response time\t|Waiting time\t|TAT time");

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

        smallest = 9;

        for(i = 0; i < limit; i++) {

            if(arrival\_time[i] <= time && temp[i] < temp[smallest] && temp[i] > 0){

                    smallest = i;

            }

        }

        if(rt[smallest] == -1){

            rt[smallest] = time - arrival\_time[smallest];

        }

        temp[smallest]--;

        if(temp[smallest] == 0) {

            count++;

            end = time + 1;

            wt = end - arrival\_time[smallest] - burst\_time[smallest];

            tat = end - arrival\_time[smallest];

            printf("\n%d\t|\t%d\t|\t%d\t|\t%d", pn[smallest], rt[smallest], wt, tat);

            wait\_time += wt;

            turnaround\_time += tat;

        }

    }

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

    average\_turnaround\_time = turnaround\_time / limit;

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

    printf("\nAverage Turnaround Time: %f", average\_turnaround\_time);

    return 0;

}

**Kết quả:**

Text

Description automatically generated

**3. Viết chương trình mô phỏng giải thuật RR với các yêu cầu sau (giả sử tất cả các tiến trình đều có arrival time là 0):**

❖ Nhập số process

❖ Nhập quantum time

❖ Nhập process name, burst time

❖ In ra Gantt chart với các thông số: process name, start processor time, stop processor time

❖ In ra average waiting time và average turnaround time

**Source code :**

#include<stdio.h>

void swap(int \*xp, int \*yp) {

    int temp = \*xp;

    \*xp = \*yp;

    \*yp = temp;

}

int main() {

    int i,j,n,time,remain,flag=0,qt;

    int pn[10],art[10],but[10],wtt[10],tat[10],rnt[10];

    float avgwt=0,avgtt=0;

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

    scanf("%d",&n);

    remain=n;

    printf("Enter quantum time: ");

    scanf("%d",&qt);

    for(i=0;i<n;i++) {

        printf("Enter process name & burst time: ");

        scanf("%d%d",&pn[i],&but[i]);

        art[i] = 0;

        rnt[i] = but[i];

    }

    for(i=0;i<n;i++) {

        for(j=i+1;j<n;j++) {

            if(art[i] > art[j]) {

                swap(pn[i], pn[j]);

                swap(art[i], art[j]);

                swap(but[i], but[j]);

                swap(wtt[i], wtt[j]);

                swap(tat[i], tat[j]);

                swap(rnt[i], rnt[j]);

            }

        }

    }

    printf("\nGantt:\n");

    printf("0");

    for(time=0,i=0;remain!=0;) {

        if(rnt[i]<=qt && rnt[i]>0) {

            time = time + rnt[i];

            printf(" -- P%d -- %d", pn[i], time);

            rnt[i]=0;

            flag=1;

        }

        else if(rnt[i] > 0) {

            rnt[i] = rnt[i] - qt;

            time = time + qt;

            printf(" -- P%d -- %d", pn[i] ,time);

        }

        if(rnt[i]==0 && flag==1) {

            remain--;

            tat[i] = time-art[i];

            wtt[i] = time-art[i]-but[i];

            avgwt = avgwt + time-art[i]-but[i];

            avgtt = avgtt + time-art[i];

            flag=0;

        }

        if(i==n-1)

            i=0;

        else if(art[i+1] <= time)

            i++;

        else

            i=0;

    }

    printf("\nAverage Waiting Time : %.2f\n",avgwt/n);

    printf("\nAverage Turnaround Time : %.2f\n",avgtt/n);

    return 0;

}

**Kết quả:**

Text

Description automatically generated