**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO BÀI TẬP NMHĐH**

**HK2, 2021-2022**

**Lab 4**

**Nhóm:** 05 **Tổ:** 01

Thành Viên Nhóm:

1: Nguyễn Trần Quang Huy (MSSV: 52000668)

**Muc lục**

[A. PHẦN THỰC HÀNH 3](#_Toc98914391)

[Ví Dụ 1: 3](#_Toc98914392)

[1.1: Code Chương Trình 3](#_Toc98914393)

[1.2: Kết Quả Demo 3](#_Toc98914394)

[Ví Dụ 2: 5](#_Toc98914395)

[2.1: Code chương trình 5](#_Toc98914396)

[2.2 Kết quả Demo 6](#_Toc98914397)

[Ví Dụ 3: 6](#_Toc98914398)

[3.1 Code chương trình 6](#_Toc98914399)

[3.2 Kết quả demo 7](#_Toc98914400)

[Ví Dụ 4: 8](#_Toc98914401)

[4.1 Code chương trình 8](#_Toc98914402)

[4.2 Kết quả demo 10](#_Toc98914403)

[Ví Dụ 5: 11](#_Toc98914404)

[5.1 Code chương trình 11](#_Toc98914405)

[5.2 Kết quả demo 13](#_Toc98914406)

[B. PHẦN BÀI TẬP 14](#_Toc98914407)

[BÀI 1: 14](#_Toc98914408)

[A: Code Chương Trình: 14](#_Toc98914409)

[B: Kết Quả Demo: 16](#_Toc98914410)

[BÀI 2: 16](#_Toc98914411)

[A: Code Chương Trình: 16](#_Toc98914412)

[B: Kết Quả Demo: 18](#_Toc98914413)

[BÀI 3: 20](#_Toc98914414)

[A: Code Chương Trình: 20](#_Toc98914415)

[B: Kết Quả Demo: 23](#_Toc98914416)

[BÀI 4: 23](#_Toc98914417)

[A: Code Chương Trình: 23](#_Toc98914418)

[B: Kết Quả Demo: 25](#_Toc98914419)

[BÀI 5: 26](#_Toc98914420)

[A: Code Chương Trình: 26](#_Toc98914421)

[Kết Quả Demo: 28](#_Toc98914422)

[BÀI 6: 29](#_Toc98914423)

[A: Code Chương Trình: 29](#_Toc98914424)

[B: Kết Quả Demo: 33](#_Toc98914425)

[KẾT LUẬN 35](#_Toc98914426)

[KẾT QUẢ LÀM VIỆC CÁC THÀNH VIÊN TRONG NHÓM 36](#_Toc98914427)

# A. PHẦN THỰC HÀNH

# B. PHẦN BÀI TẬP

# BÀI 1.1:

Tính thời gian quay vòng trung bình, thời gian chờ trung bình cho thuật toán FCFS

## A: Code Chương Trình:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "task.h"

#include "cpu.h"

#include "schedulers.h"

#define SIZE 100

static int i = 0;

Task task[SIZE];

void add(char \* name, int priority, int arrival, int burst) {

task[i].name = name;

task[i].priority = priority;

task[i].arrival = arrival;

task[i].burst = burst;

i++;

}

void schedule() {

// yc 1.2

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

for (int k = h + 1; k < i; k++) {

if (task[h].arrival > task[k].arrival) {

Task temp = task[h];

task[h] = task[k];

task[k] = temp;

}

}

}

int j = 0;

int time = 0;

float arround = 0;

float aver1;

// yc 1.1

float wait = 0;

float aver2;

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

run( & task[j], time, task[j].burst);

time += task[j].burst;

arround += time - task[j].arrival;

wait += time - task[j].burst;

}

aver1 = arround / 8;

aver2 = wait / 8;

printf("Thoi gian xoay vong trung binh cua cac tien trinh la: %.2f \n", aver1);

printf("Thoi gian cho trung binh cua cac tien trinh la: %.2f \n", aver2);

}

## B: Kết Quả Demo:

Text

Description automatically generated

# BÀI 1.2:

Giả sử thời gian đến của mỗi tiến trình khác nhau và khác 0, tính thời gian quay vòng trung bình, thời gian chờ trung bình cho thuật toán FCFS

## A: Code Chương Trình:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "task.h"

#include "cpu.h"

#include "schedulers.h"

#define SIZE 100

static int i = 0;

Task task[SIZE];

void add(char \* name, int priority, int arrival, int burst) {

task[i].name = name;

task[i].priority = priority;

task[i].arrival = arrival;

task[i].burst = burst;

i++;

}

void schedule() {

// yc 1.2

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

for (int k = h + 1; k < i; k++) {

if (task[h].arrival > task[k].arrival) {

Task temp = task[h];

task[h] = task[k];

task[k] = temp;

}

}

}

int j = 0;

int time = 0;

float arround = 0;

float aver1;

// yc 1.1

float wait = 0;

float aver2;

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

run( & task[j], time, task[j].burst);

time += task[j].burst;

arround += time - task[j].arrival;

wait += time - task[j].burst;

}

aver1 = arround / 8;

aver2 = wait / 8;

printf("Thoi gian xoay vong trung binh cua cac tien trinh la: %.2f \n", aver1);

printf("Thoi gian cho trung binh cua cac tien trinh la: %.2f \n", aver2);

}

## B: Kết Quả Demo:

Text

Description automatically generated

# BÀI 1.3:

Hiện thực tập tin schedule\_priority.c

## A: Code Chương Trình:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "task.h"

#include "cpu.h"

#include "schedulers.h"

#define MIN\_PRIORITY 1

#define MAX\_PRIORITY 10

#define SIZE 100

static int i = 0;

Task task[SIZE];

void add(char \* name, int priority, int arrival, int burst) {

task[i].name = name;

task[i].priority = priority;

task[i].arrival = arrival;

task[i].burst = burst;

i++;

}

void schedule() {

// yc 1.2

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

for (int k = 0; k < i - h - 1; k++) {

if (task[k].priority > task[k + 1].priority) {

Task temp = task[k + 1];

task[k + 1] = task[k];

task[k] = temp;

}

}

}

int j = 0;

int time = 0;

float arround = 0;

float aver1;

// yc 1.1

float wait = 0;

float aver2;

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

run( & task[j], time, task[j].burst);

time += task[j].burst;

arround += time - task[j].arrival;

wait += time - task[j].burst;

}

aver1 = arround / 8;

aver2 = wait / 8;

printf("Thoi gian xoay vong trung binh cua cac tien trinh la: %.2f \n", aver1);

printf("Thoi gian cho trung binh cua cac tien trinh la: %.2f \n", aver2);

}

## B: Kết Quả Demo:

Text

Description automatically generated

# BÀI 1.4:

Hiện thực tập tin schedule\_sjf.c

## A: Code Chương Trình:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "task.h"

#include "cpu.h"

#include "schedulers.h"

#define MIN\_PRIORITY 1

#define MAX\_PRIORITY 10

#define SIZE 100

static int i = 0;

Task task[SIZE];

void add(char \* name, int priority, int arrival, int burst) {

task[i].name = name;

task[i].priority = priority;

task[i].arrival = arrival;

task[i].burst = burst;

i++;

}

void schedule() {

// yc 1.2

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

for (int k = 0; k < i - h - 1; k++) {

if (task[k].burst > task[k + 1].burst) {

Task temp = task[k + 1];

task[k + 1] = task[k];

task[k] = temp;

}

}

}

int j = 0;

int time = 0;

float arround = 0;

float aver1;

// yc 1.1

float wait = 0;

float aver2;

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

run( & task[j], time, task[j].burst);

time += task[j].burst;

arround += time - task[j].arrival;

wait += time - task[j].burst;

}

aver1 = arround / 8;

aver2 = wait / 8;

printf("Thoi gian xoay vong trung binh cua cac tien trinh la: %.2f \n", aver1);

printf("Thoi gian cho trung binh cua cac tien trinh la: %.2f \n", aver2);

}

## B: Kết Quả Demo:

Text

Description automatically generated

# BÀI 1.5:

Hiện thực tập tin schedule\_rr.c

## A: Code Chương Trình:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "task.h"

#include "cpu.h"

#include "schedulers.h"

#define MIN\_PRIORITY 1

#define MAX\_PRIORITY 10

#define SIZE 100

static int i = 0;

Task task[SIZE];

void add(char \* name, int priority, int arrival, int burst) {

task[i].name = name;

task[i].priority = priority;

task[i].arrival = arrival;

task[i].burst = burst;

i++;

}

void schedule() {

int j = 0;

int time = 0;

int timeQuantum = 10;

while (i >= 0) {

if (task[j].burst <= timeQuantum) {

run( & task[j], time, task[j].burst);

time += task[j].burst;

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

task[h] = task[h + 1];

}

i--;

} else {

run( & task[j], time, timeQuantum);

time += timeQuantum;

task[j].burst -= timeQuantum;

}

}

}

## Kết Quả Demo:

Graphical user interface, text

Description automatically generated

# BÀI 2:

Ước lượng giá trị số PI

## A: Code Chương Trình:

#include <pthread.h>

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <sys/types.h>

#include <unistd.h>

static int counter = 0;

static int m;

void \* thr(void \* ar) {

FILE \* out\_file = fopen("m\_point.txt", "a");

int max = 2;

int min = -1;

for (int index = 0; index < m; index++) {

float x = (((float) rand() / (float)(RAND\_MAX)) \* max) + min;

float y = (((float) rand() / (float)(RAND\_MAX)) \* max) + min;

fprintf(out\_file, "(%2.2f, %2.2f)\n", x, y);

float d = sqrt(x \* x + y \* y);

if (d <= 1.0) {

counter += 1;

}

}

}

int main(int argc, char \* argv[]) {

int i;

int status, \* pstatus = & status;

int n = atoi(argv[1]);

m = atoi(argv[2]);

pthread\_t tid[n];

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

pthread\_create( & tid[i], NULL, thr, (void \* ) & tid[i]);

sleep(1);

}

printf("%d\n", counter);

printf("pi = %.6f\n", 4.0 \* counter / (n \* m));

return 0;

}

## B: Kết Quả Demo:

Text

Description automatically generated

# BÀI 3:

Ứng dụng sắp xếp bằng tiểu trình (merge sort)

## A: Code Chương Trình:

## B: Kết Quả Demo:

# KẾT LUẬN

Sau khi học và hoàn thành phần **LAB 4** nhóm thu được kết sau:

* Cách tạo các tác vụ chạy song song với nhau
* Cách tạo các tác vụ chạy gối nhau

# KẾT QUẢ LÀM VIỆC CÁC THÀNH VIÊN TRONG NHÓM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HỌ TÊN SV | MÃ SV | VAI TRÒ | CÔNG VIỆC ĐƯỢC PHÂN CÔNG | HT |
| Nguyễn Trần Quang Huy | 52000668 | TN | Làm cả phần thực hành và bài tập | 100% |
|  |  |  |  |  |

CHÚ THÍCH:

TN: TRƯỞNG NHÓM

TV: THÀNH VIÊN

HT: MỨC ĐỘ HOÀN THÀNH CÔNG VIỆC ĐƯỢC GIAO