|  |
| --- |
| Name: Nguyễn Huỳnh Nhân  ID:23521080  Class:IT007.P28.2 |

OPERATING SYSTEM  
LAB X’S REPORT

**SUMMARY**

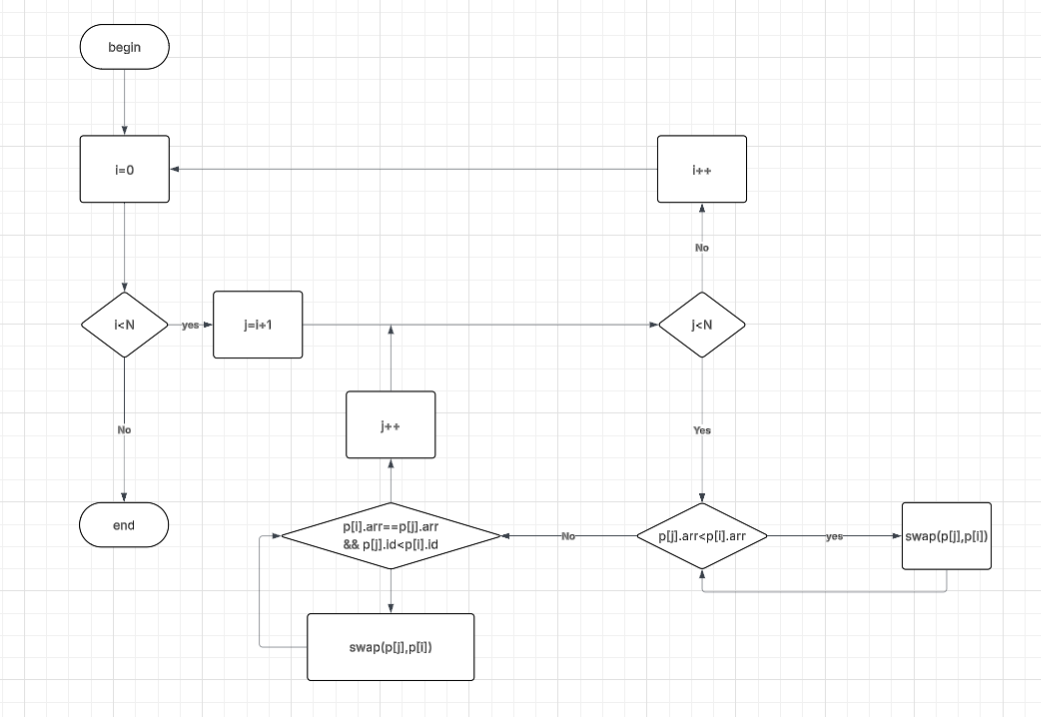
|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | | **Status** | **Page** |
| Section 4.5 | Ex 1 | Hoàn thành | 1 |
| Ex 2 | Hoàn thành | 7 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| … | … |  |  |
| … |  |  |

**Self-scrores: 10/10**

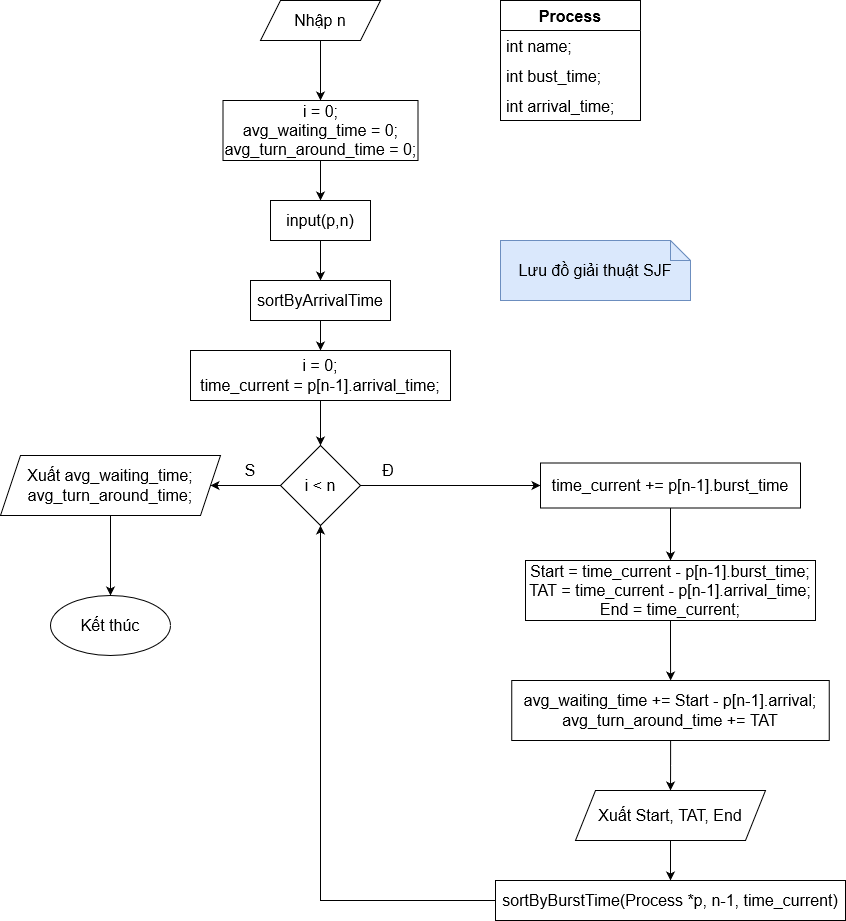
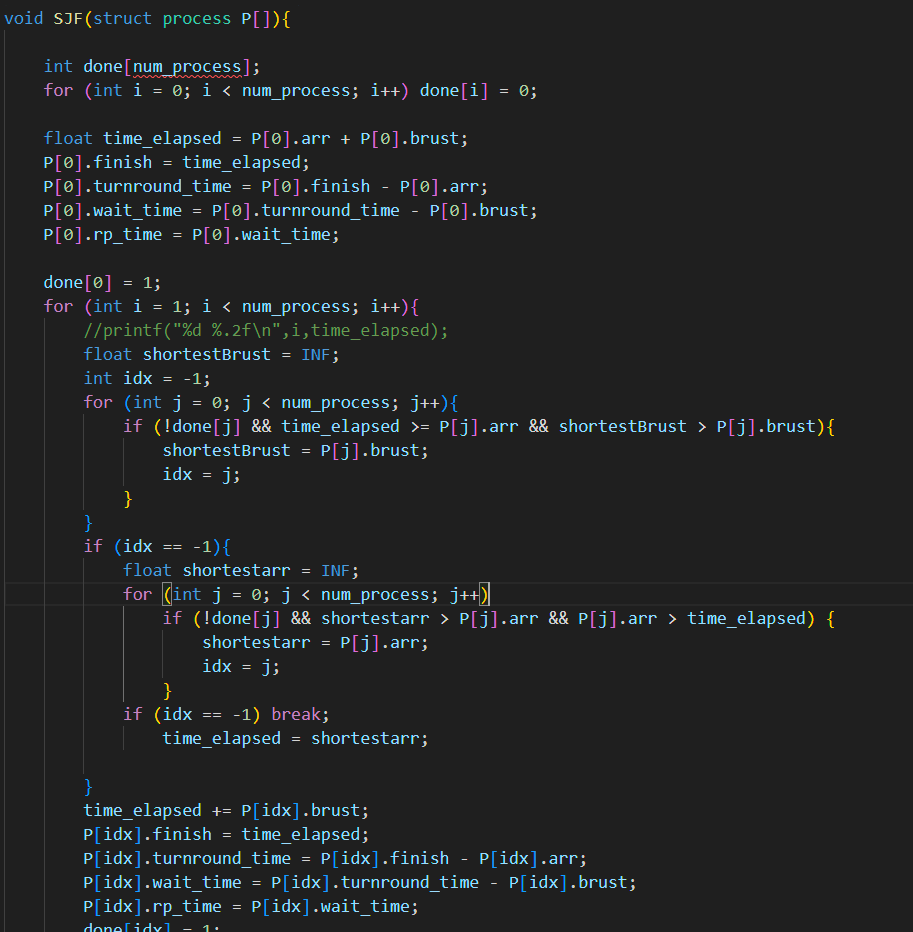
*\*Note: Export file to* ***PDF*** *and name the file by following format:* ***Student ID\_LABx.pdf***

# Section 4.5

## Task name 1: Viết chương trình mô phỏng SJF

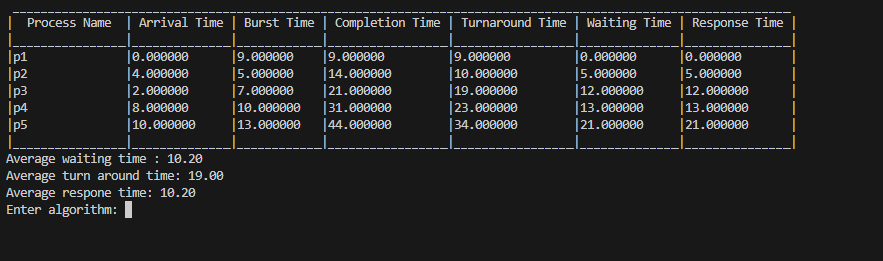
* 1. Hàm sort theo arrival time  
       
      

Giải thích :sử dụng thuật toán nổi bọt để sắp xếp tăng dần theo arrival time

* 1. Lưu đồ SJF  
     
  2. Code  
       
       
     
  3. Test case

|  |  |  |
| --- | --- | --- |
| Process 1 | Arrival Time | Burst Time |
| P1 | 0 | 9 |
| P2 | 4 | 5 |
| P3 | 2 | 7 |
| P4 | 8 | 10 |
| P5 | 10 | 13 |

Chay code

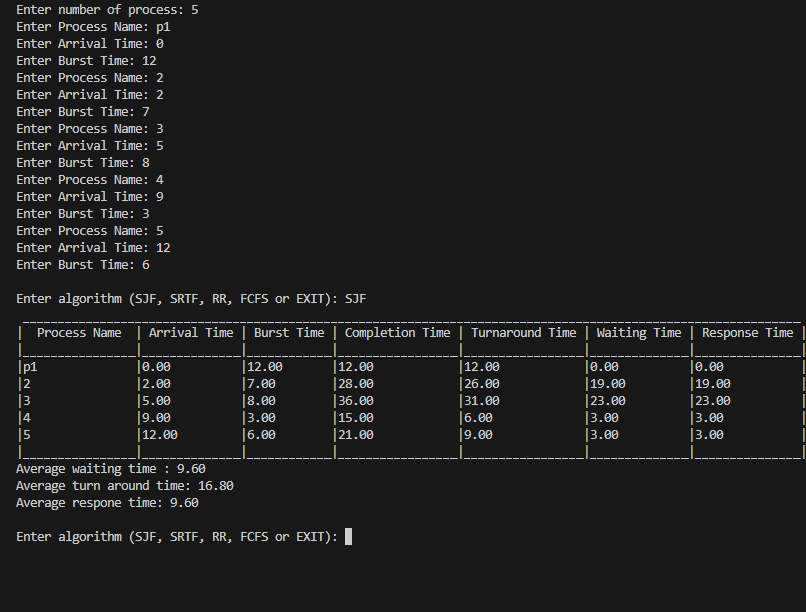


Chạy tay

Graphical user interface, application

Description automatically generated

Task2

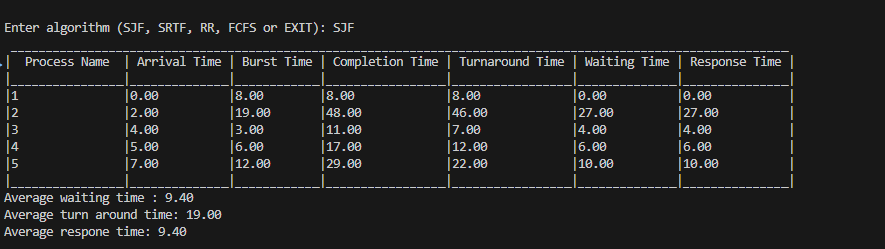


Chay tay

A picture containing text

Description automatically generated

TASK3



Chạy tay

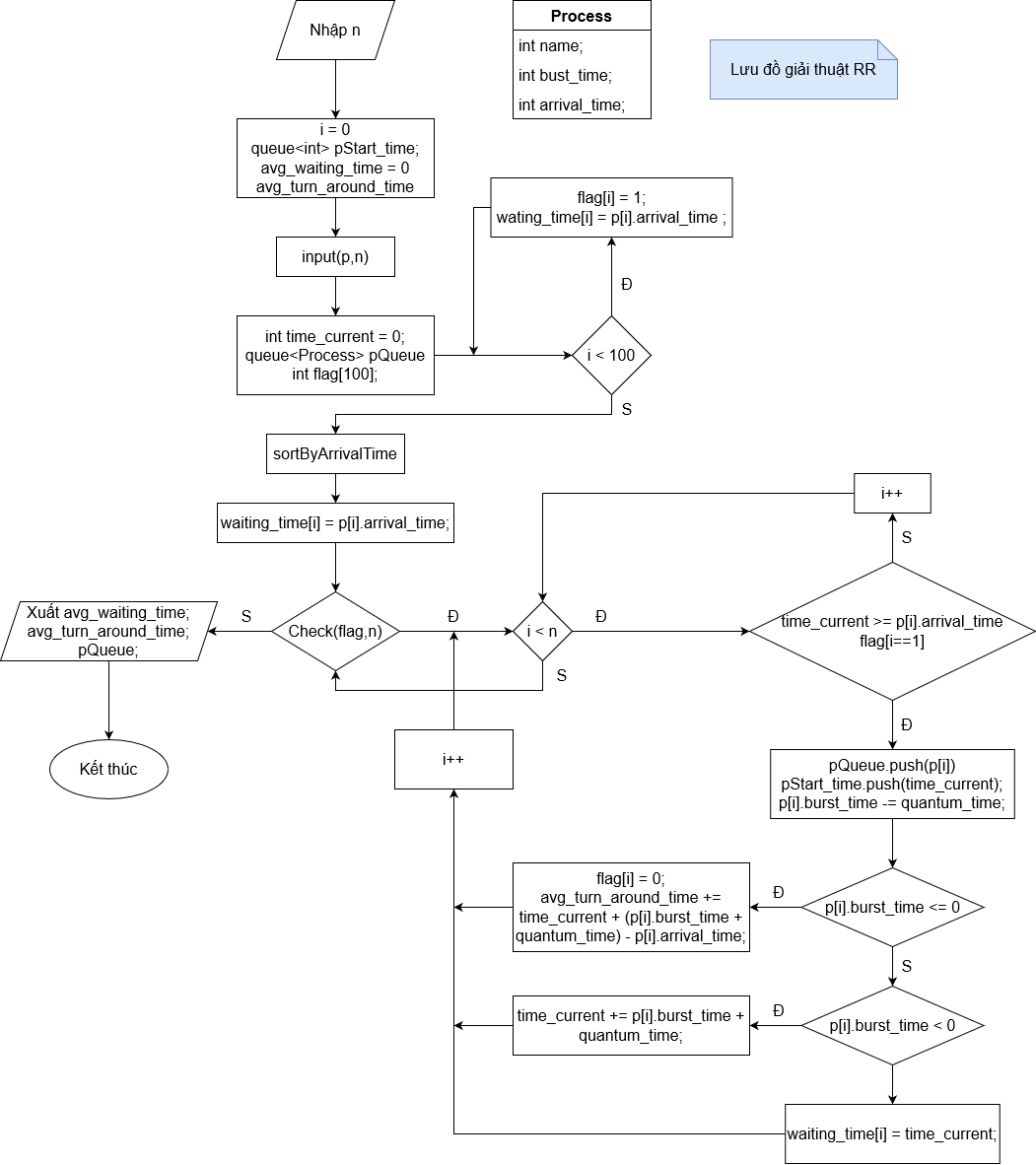
Diagram

Description automatically generated

## Task name 2:giải thuậtt RR

* 1. **Giải thuật RR**
  + **Giải thích:**
    - Tương tự như các giải thuật trên ta sẽ tiến hành tạo struct và tiến hành nhập các thông tin. Và nhập quantumtime.
    - Tạo list flag có tác dụng kiểm tra xem busrt\_time của tiến trình còn hay không. Ban đầu ta sẽ gắn hết bằng 1. Và waiting\_time sẽ bằng thời gian đến.
    - Tiến hành sort theo arrival time. Và tạo ra 1 list các tiến trình rỗng khác
    - Dùng hàm check để kiểm tra xem còn tiến trình nào vẫn còn burst\_time hay không.
    - Duyệt qua lần lượt tất cả các process. Tìn trình nào đã đến và còn burst time thì được xét. Ta sẽ tạo bản sao và được gắn vào list qQueue và các thông tin khác sẽ được lưu. Sau đó dảm burstTime đi với số lượng = quantum\_time
    - Còn nếu burst\_time == 0 thì cho flag[i] =0 và tính thời gian đợi và thười gian hoàng thành. Nếu bursttime < 0 thì thời timeline lúc này sẽ được cộng thêm burst\_time và quantum\_time. Các trường hợp khác thì time\_current là timeline lúc này được cộng thêm quantum\_time.
    - Tiến hành với các tiến trình khác và kiểm tra day flag còn phần tử nào bằng 1 hay không. Nếu hết rỏi thì xuất ra qQueue. Từ đó sẽ lấy được thông tin cụ thể.

Lưu đồ

****

**B Code**

void RR(struct process P[]) {

    float quantum;

    printf("Enter Quantum time: ");

    scanf("%f", &quantum);

    pQueue q = createQueue();

    float time\_elapsed = 0;

    int done[N] = {0};

    int numP = 0;

    int cntP = num\_process;

    while (cntP > 0) {

        while (numP < num\_process && P[numP].arr <= time\_elapsed) {

            push(q, numP++);

        }

        if (isEmpty(q)) {

            if (numP < num\_process)

                time\_elapsed = P[numP].arr;

            continue;

        }

        int idx = front(q);

        pop(q);

        if (P[idx].remain\_time == P[idx].brust)

            P[idx].rp\_time = time\_elapsed - P[idx].arr;

        float rtime = (P[idx].remain\_time < quantum) ? P[idx].remain\_time : quantum;

        time\_elapsed += rtime;

        P[idx].remain\_time -= rtime;

        while (numP < num\_process && P[numP].arr <= time\_elapsed) {

            push(q, numP++);

        }

        if (P[idx].remain\_time == 0) {

            P[idx].finish = time\_elapsed;

            P[idx].turnround\_time = P[idx].finish - P[idx].arr;

            P[idx].wait\_time = P[idx].turnround\_time - P[idx].brust;

            done[idx] = 1;

            cntP--;

        } else {

            push(q, idx);

        }

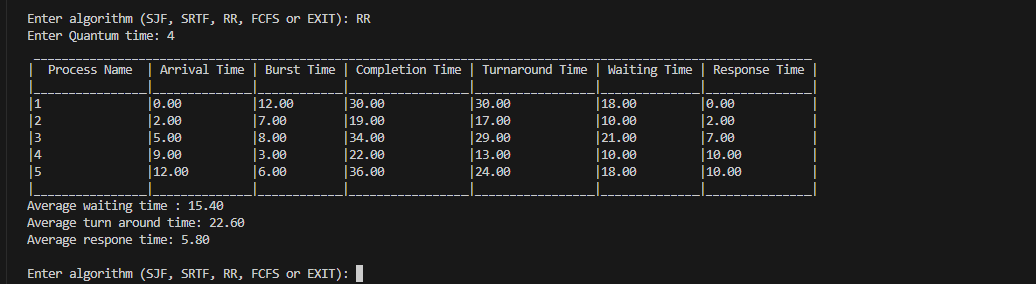
    }

    freeQueue(q);

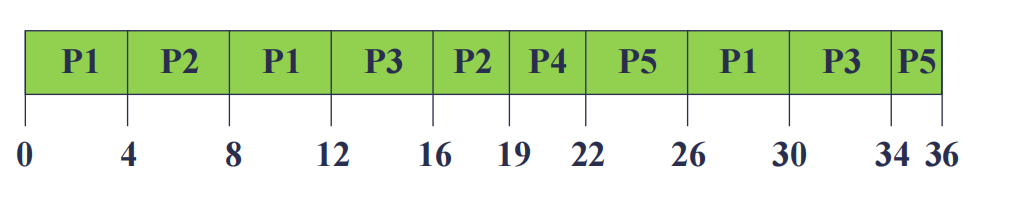
    show(P);

}

**C Test case**



Giải tay:

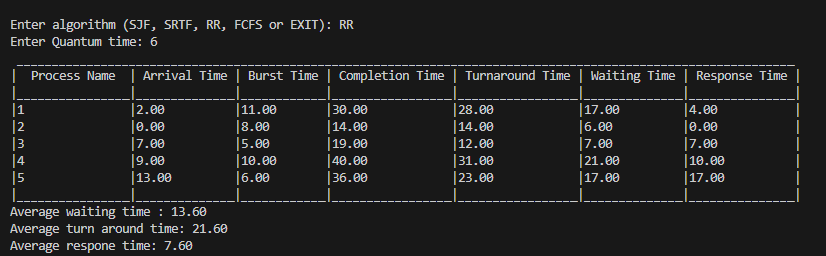


Thời gian đáp ứng trung bình: 5.8

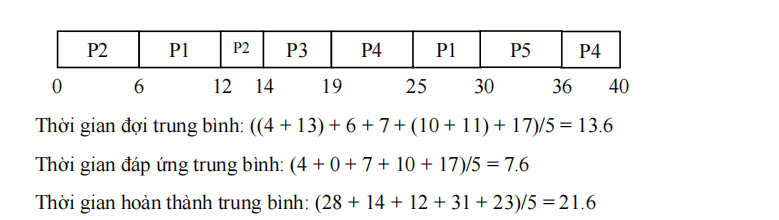
Thời gian chờ trung bình: 15.4

Thời gian hoàn thành trung bình: 22.6

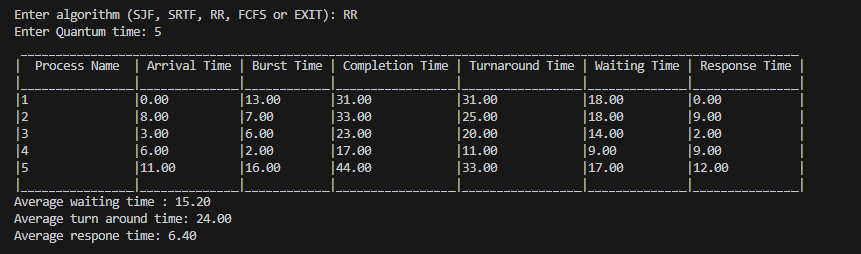
Ví dụ 2:



Chạy tay:

…

Ví dụ 3



Chạy tay

