FCFS Algorithm Project

Module : Operating System

Student : Mihoubi Youcef

Project : FCFS Algorithm

Teacher : Hammouche

Our Problem (Main Idea)

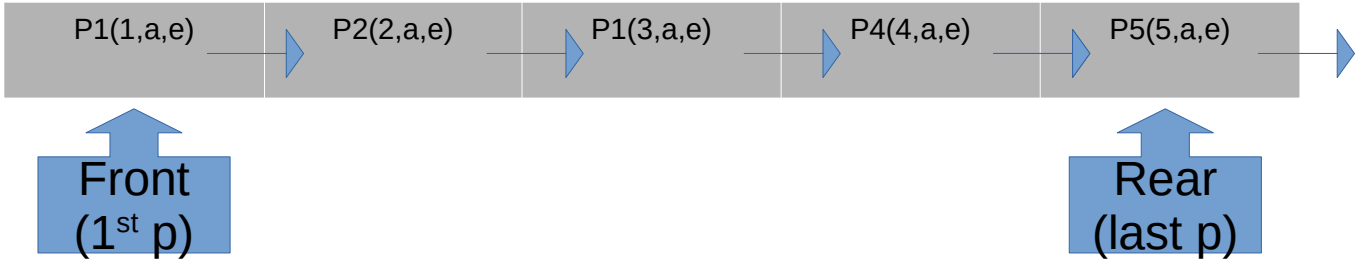
We want to create a program written by c/c++ programming language that related with simulation of FCFS CPU Algorithm (how to deal with process in First Come First served) .

We thought about a specific problem solving that gives a good result with acceptable time complexity.

Presenting The Idea

Using Queues principal (FIFO) we can say that FIFO is the same principal of FCFS just there is a different place of using

- **Structure (Process):**
- Int id (process id)
- Int arrivalTime (process arrival time) “a”
- Int executionTime (process execution time) “e”
- Process*next (that connects with next process)



Presenting The Idea

Functions :

- 1-addProcess(Enqueue)[it will create new process with it attributes (id,a,e,next pointer) then will order the queue from min (will be 1st) to max arrival time(will be last)]
- 2-deleteProcess[Dequeue a process from the list]
- 3-showFCFS[Display Queue Elements (processes) with their attributes]
- 4-computeAverages[Calculate waiting time average + turnaround average]
- 5-Main[Initializes of queue front and rear + get attributes of processes+ call the functions]

Explaining of The Algorithm

Example :

The user inputted this values :

P1 2,4	P2 0,3	P3 1,5	P4 1,1	P5 3,3	P6 4,7

The result will be :

P2 0,3	P3 1,5	P4 1,1	P1 2,4	P5 3,3	P6 4,7

Coding And Build

1- Because we're learn OS that means we should to respect the way and academic method , so we need to create cpp file using terminal commands :

joe@pop-os: ~/Desktop/se_project Mihoubi Youcef

```
joe@pop-os:~/Desktop/se_project Mihoubi Youcef$ gedit fcfs.cpp
```

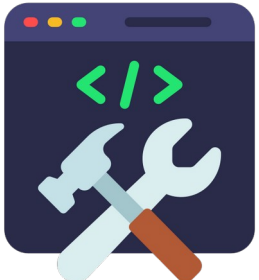
2- After the coding , we just click on save

Open — fcfs.cpp — ~/Desktop/se_project Mihoubi Youcef — Save

```cpp
#include <iostream>
using namespace std;

struct Process {
    int id;
    int arrivalTime;
    int executionTime;
    Process* next;
};

int currentTime = 0;
int totalWaitingTime = 0;
int totalTurnaroundTime = 0;
int processCount = 0;

void addProcess(Process*& front, Process*& rear, int id, int arrival, int execTime) {
    Process* newProcess = new Process();
    newProcess->id = id;
    newProcess->arrivalTime = arrival;
    newProcess->executionTime = execTime;
    newProcess->next = NULL;

    if (front == NULL || arrival < front->arrivalTime) {
        newProcess->next = front;
        front = newProcess;
        if (rear == NULL) rear = newProcess;
        return;
    }

    Process* current = front;
    while (current->next != NULL && current->next->arrivalTime <= arrival) {
        current = current->next;
    }

    newProcess->next = current->next;
    current->next = newProcess;
    if (newProcess->next == NULL) rear = newProcess;
}

void deleteProcess(Process*& front) {
    if (front != NULL) {
        Process* temp = front;
        front = front->next;
        delete temp;
    }
}

void showFCFS(Process* front) {
    cout << "\nProcess\tArrival\tExec\tWait\tTurnaround\n";
    while (front != NULL) {
```

3- we need to compile to the code

```
joe@pop-os:~/Desktop/se_project Mihoubi Youcef$ g++ fcfs.cpp -o fcfs
```



Test And Review The Program

After The Coding & Building successfully step . We need to test the program :

1.If the program opens normally.

2.If the inputs and outputs work normally.

3.If result and the application of the FCFS Algorithm are corrects.

```
joe@pop-os:~/Desktop/se_project Mihoubi Youcef$ ./fcfs
|--------------FCFS Algorithm--------------|
|---------Mihoubi Youcef L2 GRP 10---------|
Enter the number of processes: 4
Enter arrival time and execution time for Process 1 (separated by space): 2 3
Enter arrival time and execution time for Process 2 (separated by space): 1 4
Enter arrival time and execution time for Process 3 (separated by space): 0 7
Enter arrival time and execution time for Process 4 (separated by space): 1 2

FCFS :
Process Arrival Exec    Wait    Turnaround
P3      0       7       0       7
P2      1       4       6       10
P4      1       2       10      12
P1      2       3       11      14

Average Waiting Time: 6.75
Average Turnaround Time: 10.75
joe@pop-os:~/Desktop/se_project Mihoubi Youcef$
```

Result

The implemented FCFS scheduling algorithm successfully simulates the behavior of a real CPU queue using a linked list structure. By processing and removing each node in order, the program respects the true FIFO principle. The modular functions (enqueue, dequeue, display, and compute averages) provide clarity and efficiency with acceptable complexity. This implementation offers a solid foundation for understanding basic scheduling concepts in operating systems