

**DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN**

# Sistemas Operativos

Laboratorio 5

Planificación de CPU

# INTRODUCCIÓN

La planificación de la CPU aborda el problema de decidir cuál de los procesos en la cola de procesos listos se asignará al núcleo de la CPU.

La planificación de la CPU es la base de los sistemas operativos multiprogramados. Al cambiar la CPU entre procesos, el sistema operativo puede hacer que la computadora sea más productiva. En esta parte se introducirán conceptos básicos de planificación de la CPU y se presentarán varios algoritmos de planificación de la CPU.

# OBJETIVOS

El objetivo de este laboratorio es proporcionar una comprensión de algoritmos de planificación de la CPU. Se aplicarán técnicas de modelado y simulación para evaluar los algoritmos de planificación de la CPU, y se diseñarán programas que implementen estos algoritmos, por tanto, se espera que el alumno pueda comprender los diversos mecanismos relacionados con la planificación de CPU.

# ANTECEDENTES: FIRST-COME-FIRST-SERVED

El algoritmo de planificación de CPU más sencillo es el algoritmo FCFS. Se trata de un algoritmo NO APROPIATIVO. Esto implica que, una vez que se asigna un proceso al procesador, este lleva a cabo todas sus instrucciones hasta su finalización. La planificación FCFS se ajusta en el sentido formal o humano con la equidad, pero es injusta en el sentido de que las tareas largas hacen esperar a las tareas cortas y las tareas no importantes hacen esperar a las tareas importantes.

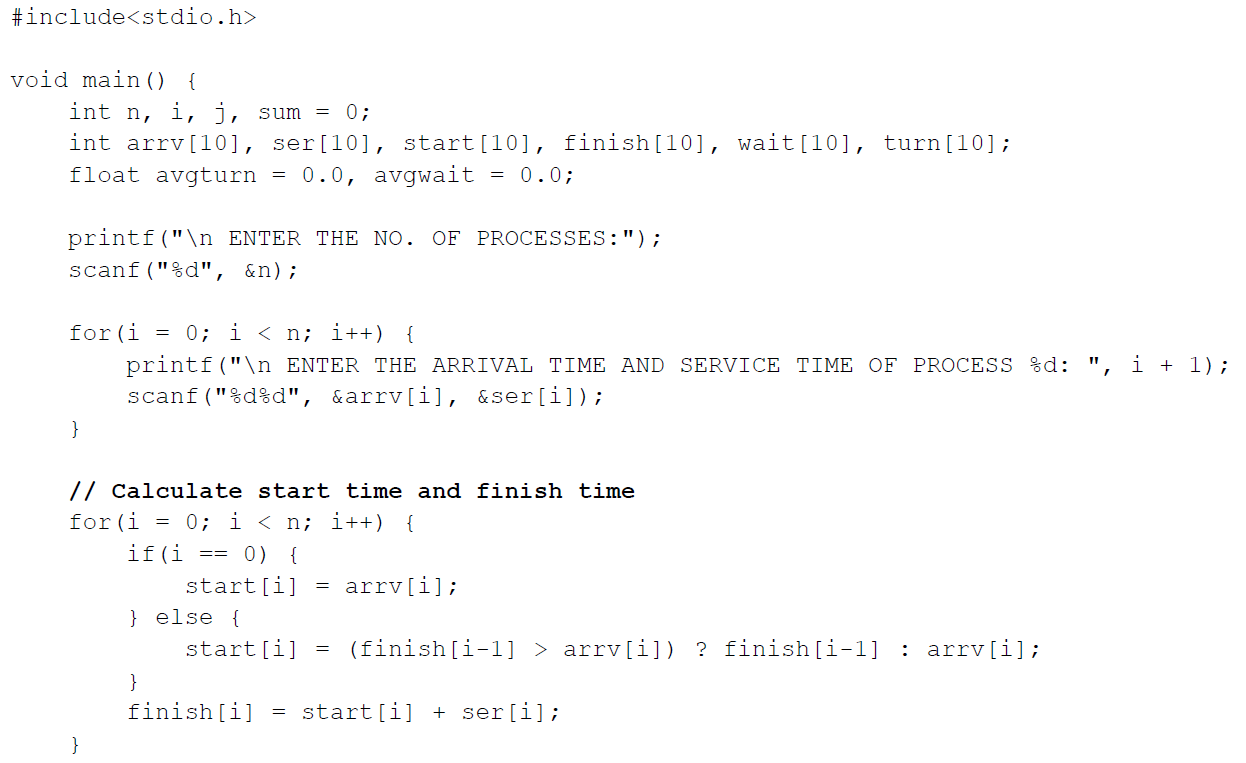
Para la representación gráfica de los procesos, se emplea una tabla. El eje Y corresponde a los procesos, y el eje X representa los ciclos de CPU (tiempo de ejecución).

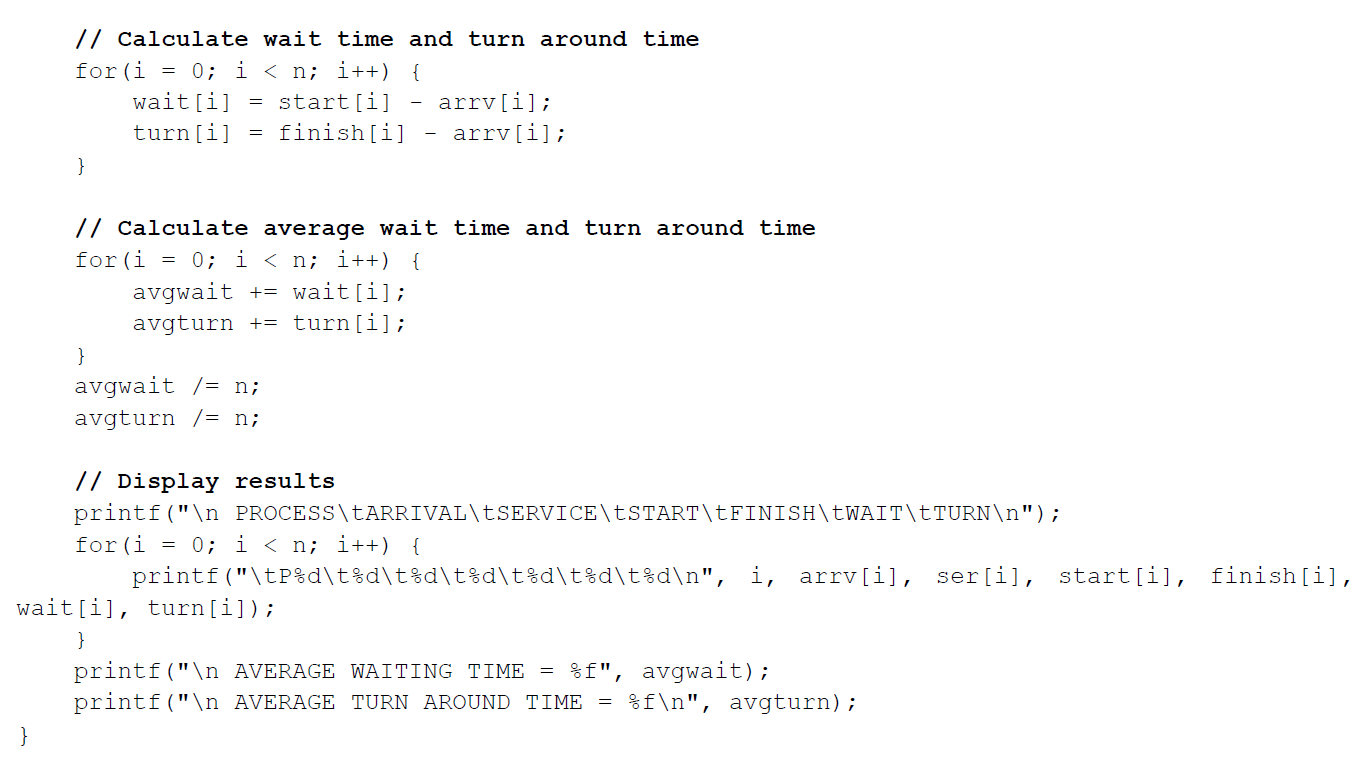
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

**Ejercicio 1:** Dada la siguiente carga de procesos y código, determinar el tiempo medio de espera y tiempo medio de retorno. **20pts**

**Carga de procesos**

|  |  |  |
| --- | --- | --- |
| **Proceso** | **Llegada** | **Ciclos de CPU** |
| E | 6 | 3 |
| D | 5 | 5 |
| C | 3 | 4 |
| B | 2 | 3 |
| A | 0 | 6 |





Texto

Descripción generada automáticamente

Después de un arreglo rápido del código en algunos componentes se da a conocer que, para esta tabla de datos, el tiempo promedio de espera es de 13.6 unidades y el tiempo de medio de retorno es de 17.79 unidades.

# ANTECEDENTES: SHORTEST JOB FIRST

El algoritmo SJF asigna el proceso que tenga menor tiempo de CPU al procesador. Al igual que en el caso anterior (FCFS), se trata de un algoritmo NO APROPIATIVO. Esto quiere decir que una vez que el proceso ha tomado el procesador, no puede retirársele hasta que este haya finalizado.

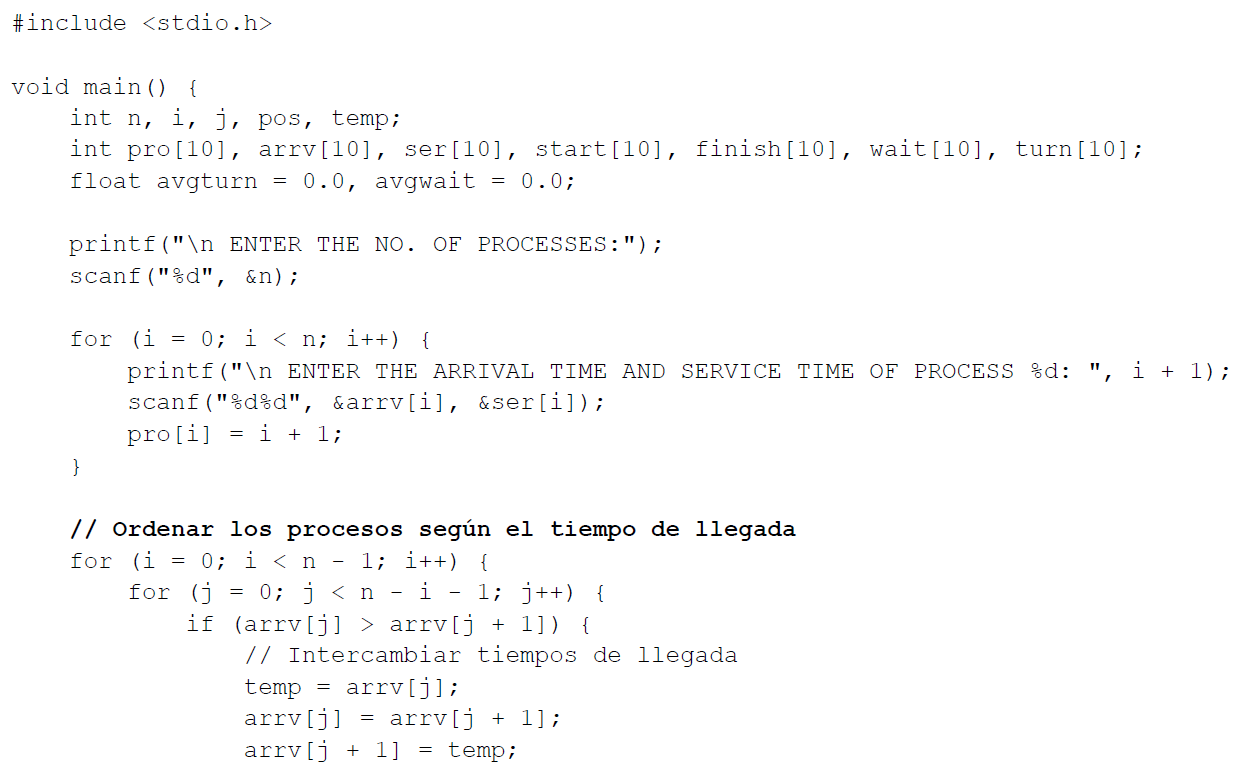
El algoritmo SJF favorece trabajos cortos a expensas de los más largos. El problema obvio con el esquema SJF es que requiere un conocimiento preciso de cuánto tiempo va a tardar un trabajo o proceso, y esta información no suele estar disponible. Lo mejor que puede hacer el algoritmo es confiar en las estimaciones de tiempo de ejecución proporcionadas por el usuario.

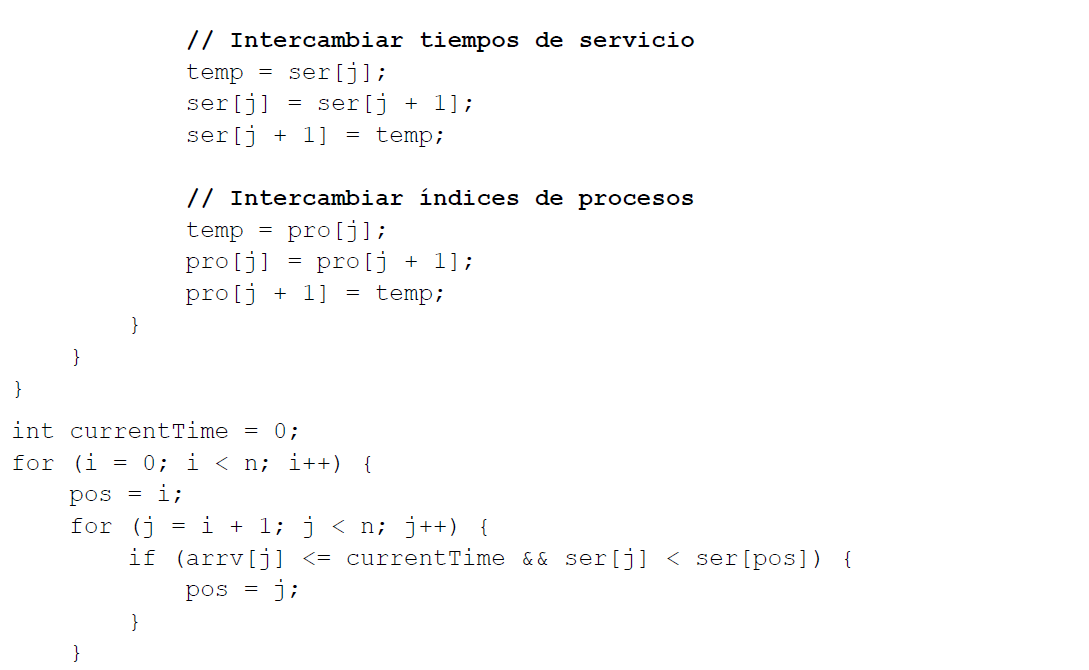
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

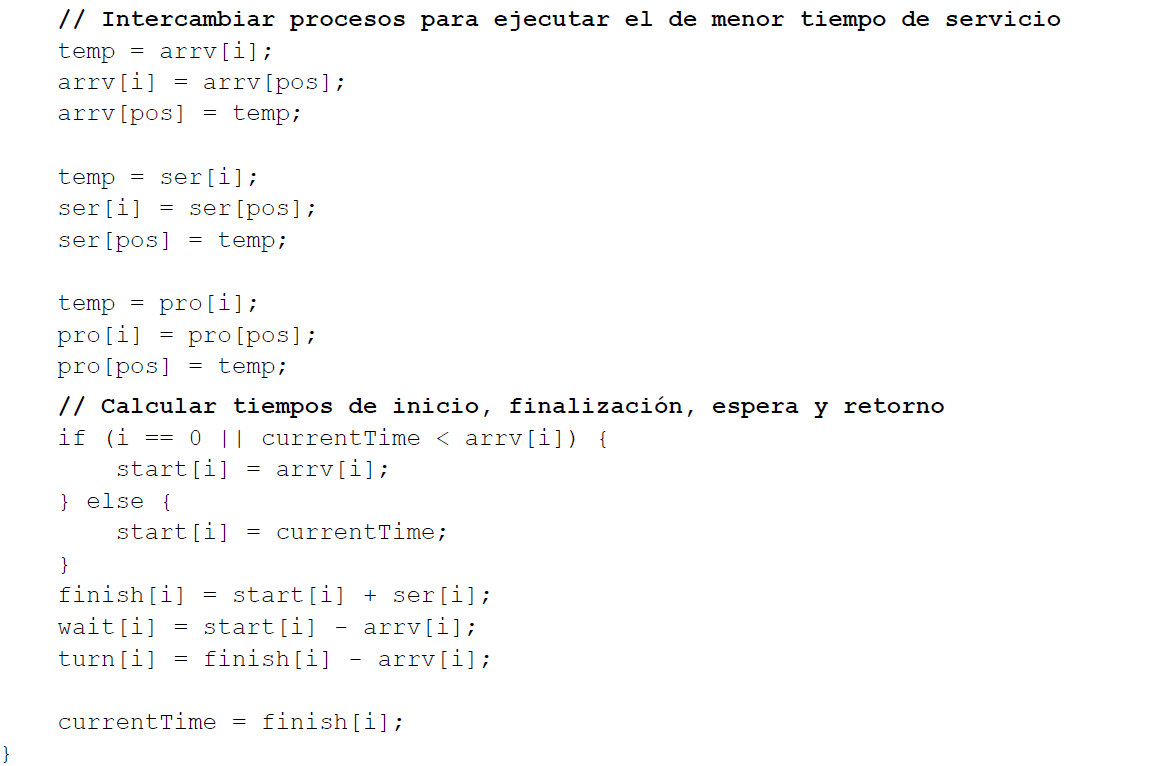
**Ejercicio 2:** Dada la siguiente carga de procesos y código, determinar el tiempo medio de espera y tiempo medio de retorno. Completar tabla de colores según secuencia obtenida. **30pts**

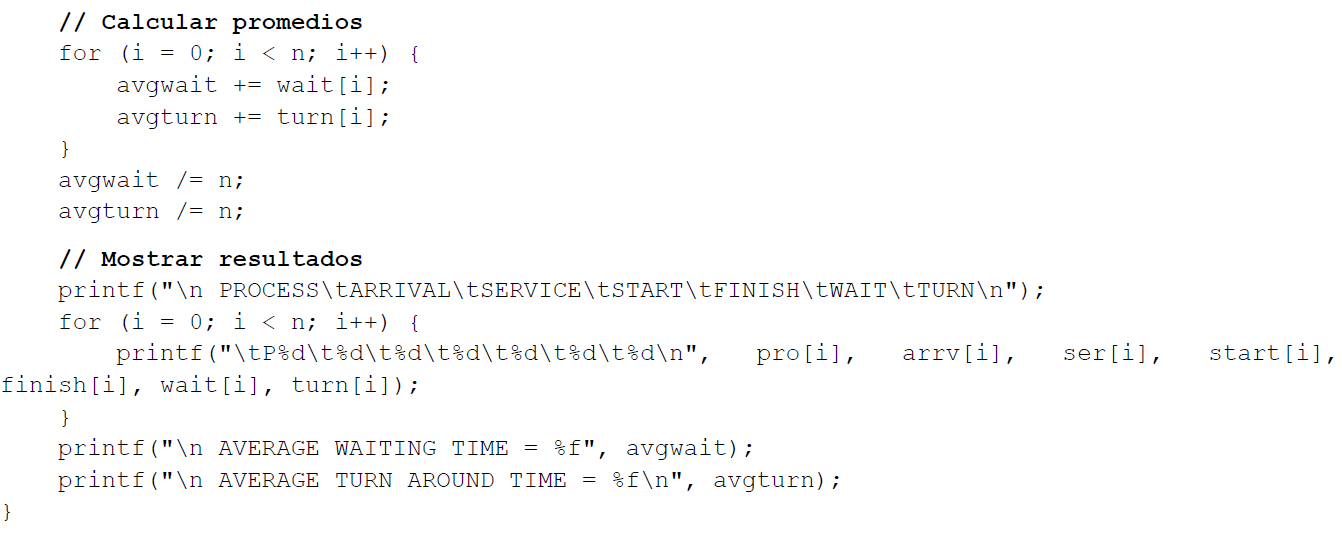
**Carga de procesos**

|  |  |  |
| --- | --- | --- |
| **Proceso** | **Llegada** | **Ciclos de CPU** |
| E | 6 | 3 |
| D | 5 | 5 |
| C | 3 | 4 |
| B | 2 | 3 |
| A | 0 | 6 |







****

**Pantalla de computadora con letras

Descripción generada automáticamente con confianza media**

Como podemos ver en la imagen adjunta, los resultados de unidades de tiempo si son mejores que los anteriores, y tal como se esperaba, en la tabla de colores se ve reflejado que los procesos con menos tiempo se hicieron primero, aunque los que llegaron antes tuvieron que esperar para poder realizarse.

# ANTECEDENTES: SCHEDULING ROUND ROBIN

Se trata de un algoritmo apropiativo que va asignando a cada uno de los procesos un intervalo de tiempo de CPU predefinido (Quantum). Cuando un proceso recibe la CPU y agota su quantum, debe abandonar el procesador. También puede abandonar la CPU sin agotar el quantum de tiempo. Esto último sucederá cuando el proceso termine su ejecución.

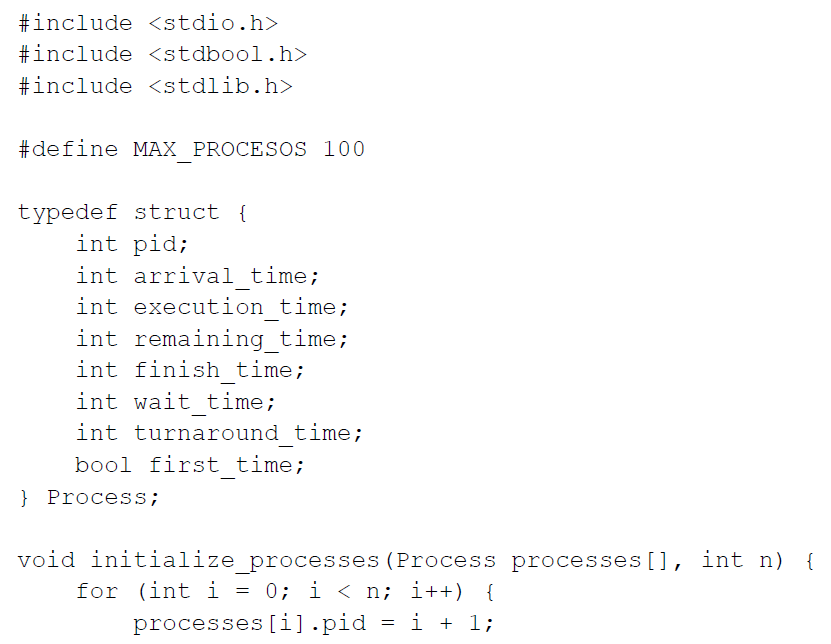
Al ser un algoritmo apropiativo es efectivo en entornos de compartición de tiempo en los que el sistema necesita garantizar tiempos de respuesta razonables para los usuarios interactivos. La única cuestión interesante con el esquema Round Robin es la longitud del quantum. Establecer el quantum demasiado corto provoca demasiados cambios de contexto y reduce la eficiencia de la CPU. Por otro lado, establecer el quantum demasiado largo puede causar un mal tiempo de respuesta y se aproxima al FCFS.

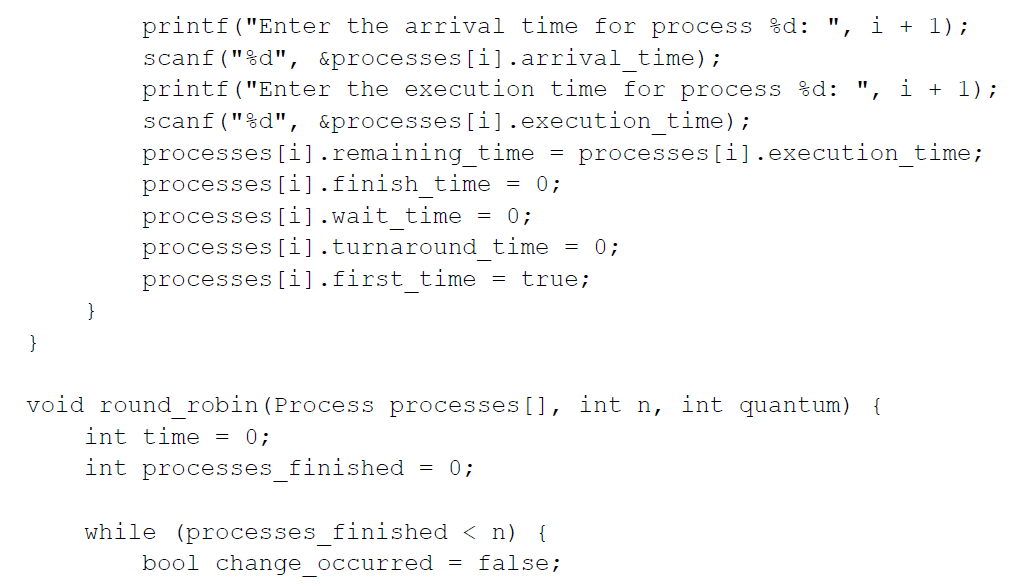
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

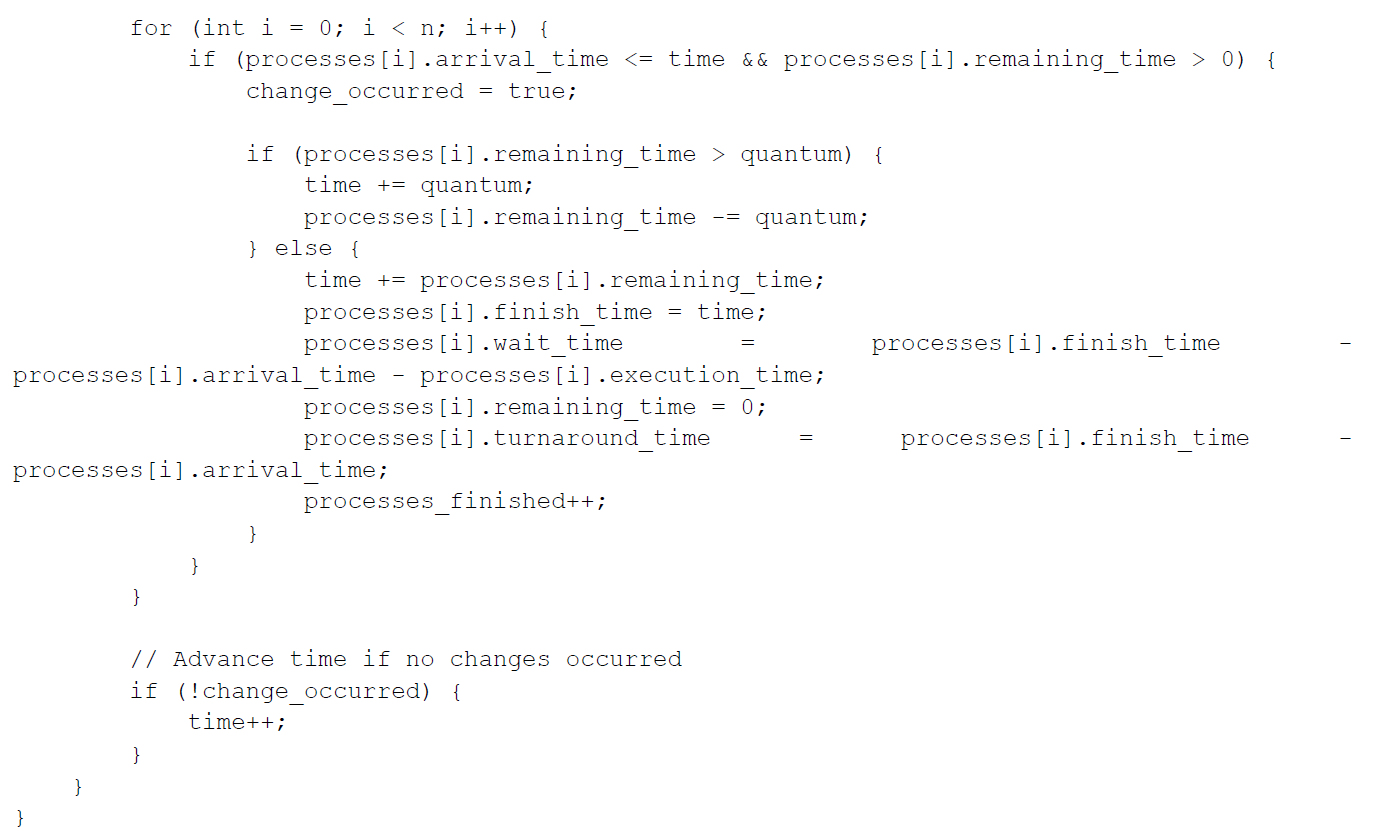
**Ejercicio 3:** Dada la siguiente carga de procesos y código, determinar el tiempo medio de espera, tiempo medio de retorno con un quantum = 4 y completar la tabla Gantt según la secuencia. **40pts**

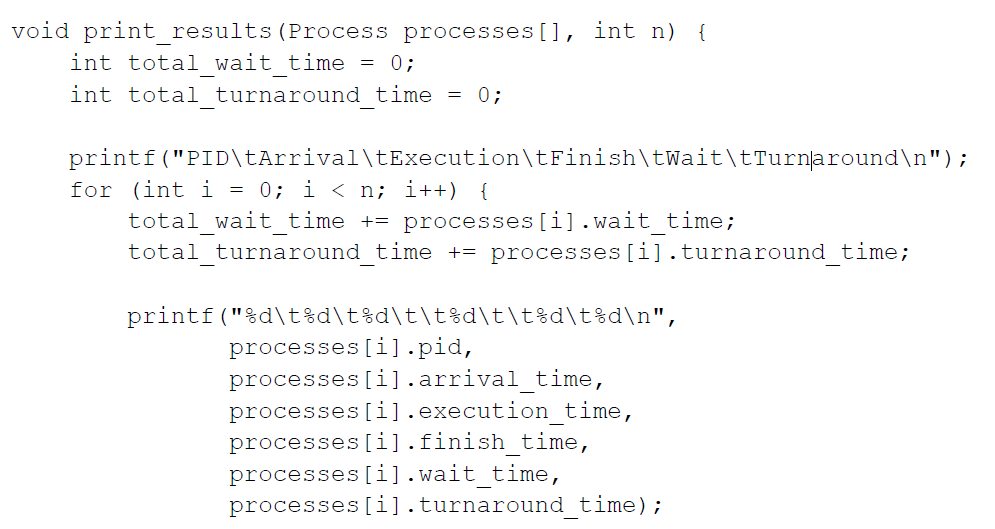
**Carga de procesos**

|  |  |  |
| --- | --- | --- |
| **Proceso** | **Llegada** | **Ciclos de CPU** |
| E | 4 | 3 |
| D | 3 | 5 |
| C | 3 | 4 |
| B | 2 | 3 |
| A | 0 | 6 |







0

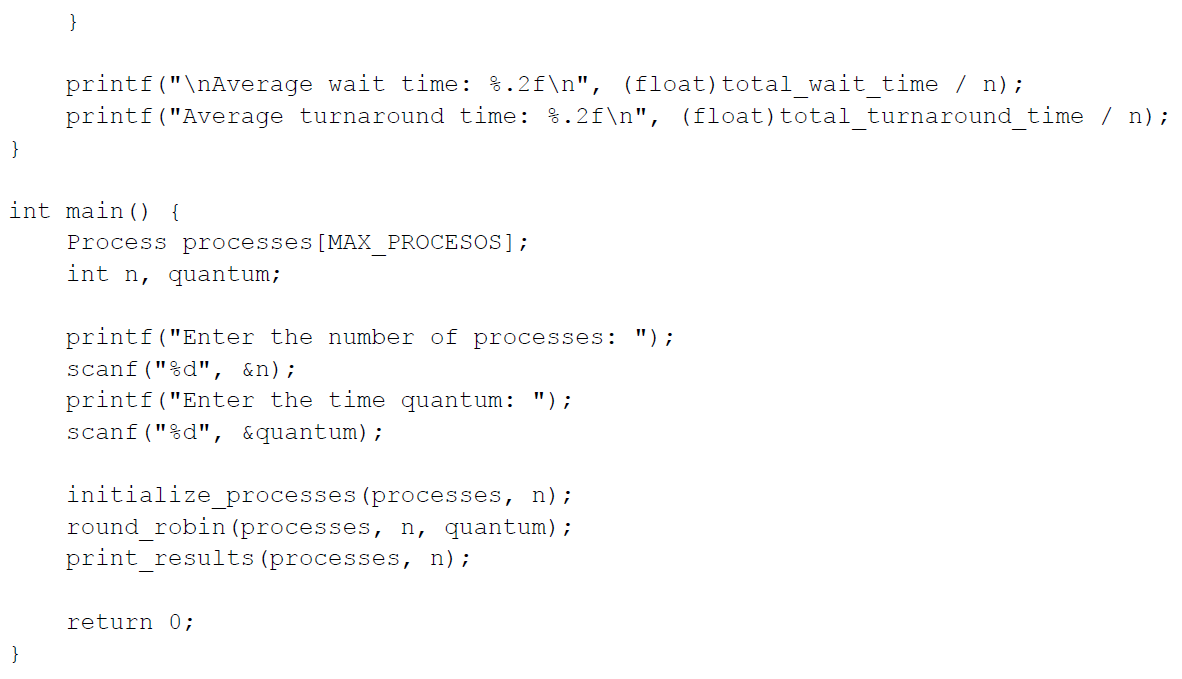


Imagen que contiene Texto

Descripción generada automáticamente

En la imagen adjunta se pueden ver los resultados obtenidos del código, aun que no son muy representativos el diagrama Gantt representa mejor la distribución implementada.

# Entrega (10 pts)

Documente (screenshots) cada una de las acciones antes señaladas. Es de exclusiva responsabilidad del estudiante respetar el formato de entrega de informe de esta guía **(debajo de cada enunciado su screenshot en donde aparezca de forma clara las sentencias utilizadas)**. El formato de entrega debe ser en PDF, y el nombre del archivo debe contener su nombre y apellido (**Laboratorio\_5\_Nombre\_Apellido**). Todas las actividades deben ser entregadas (subidas) a la plataforma digital en las fechas establecidas, por cada hora de atraso, se descontará 1 pto.