UNIVERSIDAD DE LOS LLANOS
Facultad de Ciencias básicas e ingenierías

INFORME DE LABORATORIO SISTEMAS OPERATIVOS

ALGORITMOS DE PLANIFICACIÓN DE PROCESOS

B. Rafael Durán [1], J. Pablo López[2]

1. Cod: 160004510, Ing. Sistemas
2. Cod: 160004212, Ing. Sistemas

Facultad de Ciencias Básicas e Ingenierías.
Ingeniería de Sistemas

Resumen

En el presente artículo, se evidencia el análisis, desarrollo y codificación de 3 de los algoritmos de planificación de procesos los cuales son FCFS (First Come, First Served), SJF (Shortest Job First), prioridad y round Robin, cada uno tiene su particularidad el cual se observan en el desarrollo de la práctica de laboratorio, para ello se tuvo en cuenta variables como el tiempo de CPU, prioridad, número de procesos, tiempo de espera, Quantum y se debe visualizar en la ejecución del programa el tiempo promedio de espera y el diagrama de Gantt.

Palabras clave: Algoritmos de planificación de procesos, FCFS, SJF, prioridad, diagrama de Gantt.

1. Introducción

El siguiente informe tiene como objetivos dar a conocer los algoritmos de planificación de procesos lo cuales son FCFS, SJF, Prioridad y Round Robin para ello se implementará mediante el entorno de desarrollo integrado de código abierto codeblocks, se hará uso de múltiples variables las cuales se observarán a lo largo del análisis, cada parte es fundamental, pero esta en específico es la base de todo, no se puede construir código de la nada, primero hay que definir los diagramas de entrada/salida y basándose en ellos sale la navegación del programa mediante los diagramas de flujo.

2. Referente teórico

Un proceso es básicamente un programa en ejecución y tiene 3 estados en los que se puede encontrar, que son los siguientes: "listo","bloqueado" y "en ejecución". Los sistemas operativos cuentan con un componente llamado planificador, que se encarga de decidir cuál de los procesos hará uso del procesador. La toma de esta decisión, así como el tiempo de ejecución del proceso, estará dada por un algoritmo, denominado Algoritmo de Planificación.

Planificación de procesos en un S.O (Sistema Operativo).

Conjunto de políticas y mecanismos incorporados al sistema operativo, a través de un módulo denominado planificador, que debe decidir cuál de los procesos en condiciones de ser ejecutado conviene ser despachado primero y qué orden de ejecución debe seguirse. Esto debe realizarse sin perder de vista su principal objetivo que consiste en el máximo aprovechamiento del sistema, lo que implica proveer un buen servicio a los procesos existentes en un momento dado.

Objetivos de la planificación de procesos.

Equidad, ya que todos los procesos deben ser atendidos.

Eficacia, el procesador debe ser ocupado al 100%.

Tiempo de respuesta, el tiempo empleado en dar respuesta a las solicitudes del usuario debe ser lo más corto posible.

Rendimiento, maximizar el número de procesos en determinado tiempo.

Algoritmos de planificación de procesos.

Existen muchos algoritmos de planificación de procesos, los cuales se observarán en próximas prácticas de laboratorios, para la práctica actual se visualizarán los 3 siguientes:

FCFS (FIRST COME, FIRST SERVED).

Primero en llegar, primero en ser servido. Este algoritmo emplea una estructura de datos muy típica, la cola, el primero en llegar es el primero en salir o en ser "servido". Cuando un proceso comienza a ejecutarse no termina hasta que este sea finalizado.

SJF (SHORTEST JOB FIRST).

El trabajo más corto primero, El proceso que se encuentra en ejecución cambiará de estado voluntariamente, o sea, no tendrá un tiempo de ejecución determinado para el proceso. A cada proceso se le asigna el tiempo que usará cuando vuelva a estar en ejecución, y se irá ejecutando el que tenga un menor tiempo asignado. Si se da el caso de que dos procesos tengan igual valor en ese aspecto emplea el algoritmo FCFS.

PLANIFICACIÓN POR PRIORIDAD.

En este tipo de planificación, se le da una prioridad a cada proceso siguiendo un criterio determinado y de acuerdo a esa prioridad será el orden en el que se atiende cada uno. **[1]**

ROUND ROBIN.

Este es uno de los algoritmos más antiguos, sencillos y equitativos en el reparto de la CPU entre los procesos, muy válido para entornos de tiempo compartido. Cada proceso tiene asignado un intervalo de tiempo de ejecución, llamado **Quantum**. Si el proceso agota su *Quantum* de tiempo, se elige a otro proceso para ocupar la CPU. Si el proceso se bloquea o termina antes de agotar su *quantum* también se alterna el uso de la CPU. El *round robin* es muy fácil de implementar. Todo lo que necesita el planificador es mantener una **lista** de los procesos listos.

3. Sección experimental

Análisis y diseño.

Para esta parte del desarrollo experimental se comienza con la realización de los diagramas de entrada/salida de cada uno de los algoritmos los cuales vamos a ver por simulación en la práctica de laboratorio actual. Esto con el objetivo de mostrar una aproximación inicial de los datos y variables que se utilizarán a la hora de implementar el algoritmo de programación, donde entrada son los datos con los cuales cuenta el programa que requiere cada planificación de proceso (Tiempo de CPU, número de procesos, tiempo de llegada y prioridad) Y la salida es el resultado/solución el cual se le da al problema (que para este caso es el diagrama de Gantt y el tiempo promedio de espera). En la figura 1,2 y 3 que se muestran a continuación se observan los diagramas de entrada/salida:

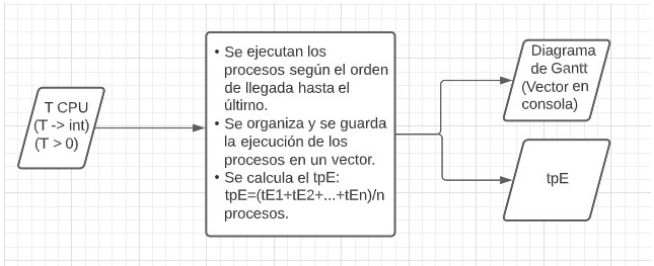


Figura 1. Diagrama de entrada/salida algoritmo FCFS. Autores: Brian R. Duran, Juan P. Lopez.

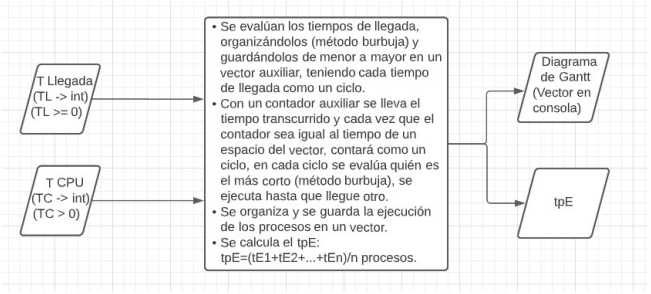


Figura 2. Diagrama de entrada/salida algoritmo SJF. Autores: Brian R. Duran, Juan P. Lopez.

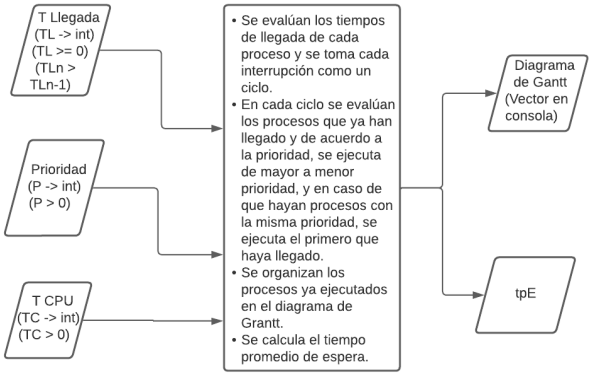


Figura 3. Diagrama de entrada/salida algoritmo de prioridad. Autores: Brian R. Duran, Juan P. Lopez.

ALGORITMOS DE PLANIFICACIÓN DE PROCESOS

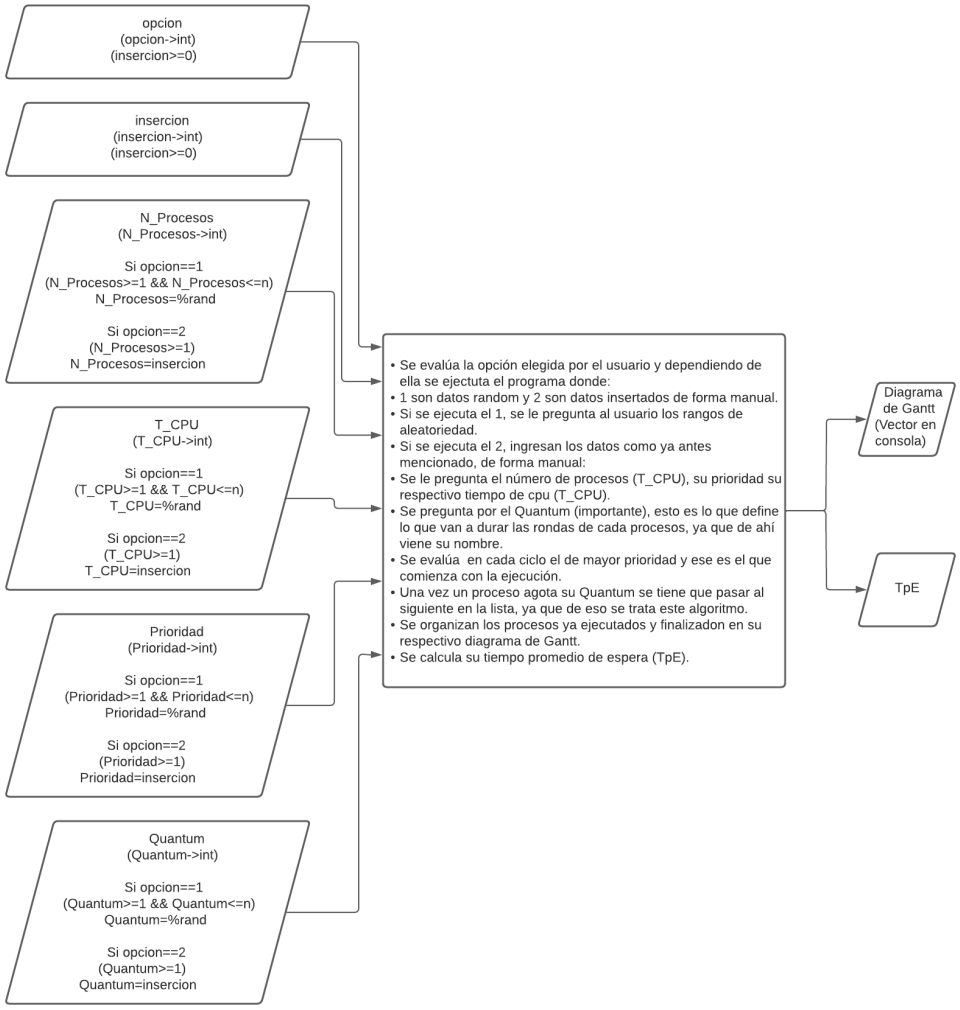
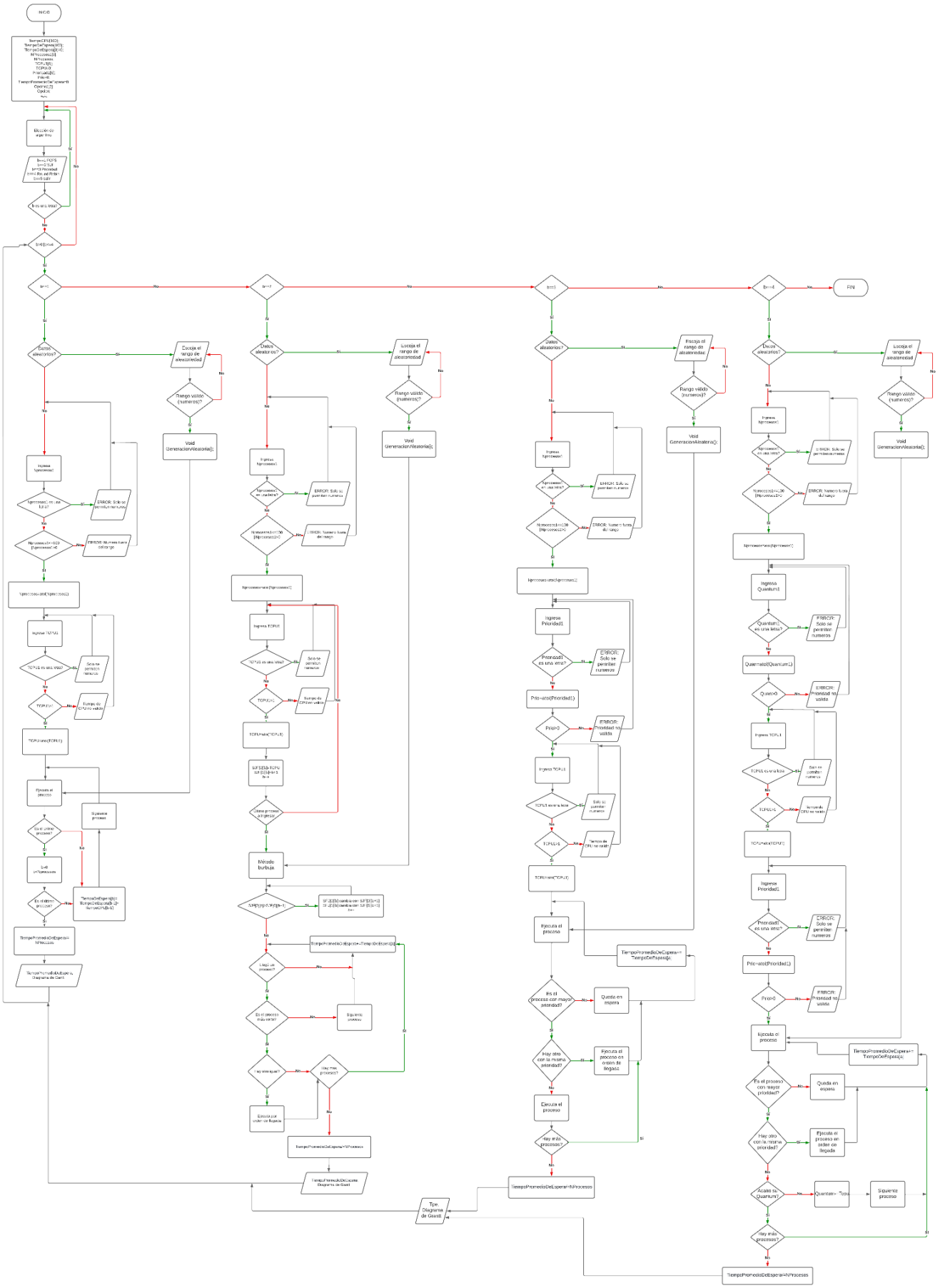


Figura 4. Diagrama de entrada/salida algoritmo de Round Robin. Autores: Brian R. Duran, Juan P. Lopez.

Una vez terminados los diagramas de entrada y salida se procede a realizar la parte de los diagramas de flujo, el cual indicará la navegación que realizará el algoritmo a través de la simulación. El diagrama de flujo describe a grandes rasgos el desarrollo de los algoritmos de planificación de procesos, tal y como se muestra en la figura 4 a continuación:

ALGORITMOS DE PLANIFICACIÓN DE PROCESOS



ALGORITMOS DE PLANIFICACIÓN DE PROCESOS

Figura 5. Diagrama de flujo del programa completo. Autores: Brian R. Duran, Juan P. Lopez. Hecho en: https://lucid.app/lucidchart/f26104db-0d55-4d15-85b4-4f34aeb7edb7/edit?viewport_loc=288%2C1641%2C2595%2C1262%2C0_0&invitationId=inv_ea46a85e-22c9-4d9a-995d-7218bae86996#

Para esta parte del desarrollo experimental se hizo uso de la herramienta web Lucidchart:

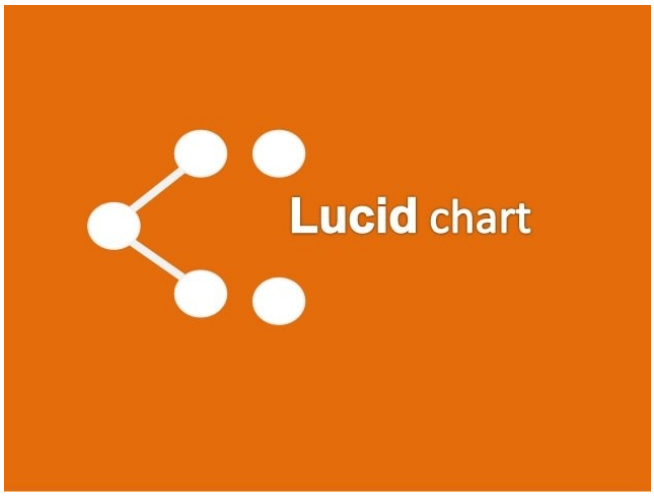


Figura 6. LucidChart, herramienta web utilizada para la realización de diagramas de flujo y de entrada/salida.**[2]**

Codificación.

Para la codificación de los algoritmos la cual será presentada a continuación, se utilizó el lenguaje de programación c++ y el entorno de desarrollo integrado de código abierto Code::Blocks:

Figura 7. Compilador Code::Blocks.**[3]**

La figura 8, 9 y 10 a continuación muestra el funcionamiento del diagrama de Gantt de cada algoritmo de planificación de procesos respectivamente en la simulación a partir de la herramienta Code::Blocks:

```
cout<<"----------DIAGRAMA DE GANTT----------\n\n";
cout<<"       0 ms."<<endl;
for(int a=0,b=0;a<NProcesos;a++){
    cout<<"        ^\n        |\n        P"<<a+1<<"\n        |\n        v"<<endl;
    b+=TiempoCPU[a];
    cout<<"        "<<b<<" ms."<<endl;
}
```

Figura 8. Codificación diagrama de Gantt algoritmo FCFS.

```
cout<<"\nEl tiempo promedio de espera es de: "<<TiempoPromedioDeEspera<<" ms.\n"<<endl;
cout<<"----------DIAGRAMA DE GANTT----------\n\n";
cout<<"       0 ms."<<endl;
for(int a=0,b=0;a<NProcesos;a++){
    cout<<"        ^\n        |\n        P"<<SJF[1][a]<<"\n        |\n        v"<<endl;
    b+=SJF[0][a];
    cout<<"        "<<b<<" ms."<<endl;
}
```

Figura 9. Codificación diagrama de Gantt algoritmo SJF.

```
cout<<"\nEl tiempo promedio de espera es de: "<<TiempoPromedioDeEspera<<" ms.\n"<<endl;
cout<<"---------DIAGRAMA DE GANTT---------\n\n";
cout<<"       0 ms."<<endl;
for(int a=0,b=0;a<NProcesos;a++){
    cout<<"        ^\n        |\n        P"<<Prioridad[1][a]<<"\n        |\n        v"<<endl;
    b+=Prioridad[2][a];
    cout<<"        "<<b<<" ms."<<endl;
}
```

Figura 10. Codificación diagrama de Gantt algoritmo de prioridad.

4. Resultados

Pruebas e implementación.

Al momento de completar todo el análisis y el diseño se procede a realizar las respectivas pruebas de escritorio, esto con el fin de hallar errores y comprobar el funcionamiento de los algoritmos e ir corrigiendo cada error conforme se avanza en la codificación.

Para comenzar, prueba de escritorio del algoritmo FCFS:

n_proceso	TCPU	Tesp
1	2	0
2	3	2
3	4	5
4	5	9
Tpe:		4 ms

Tabla 1. Prueba de escritorio algoritmo FCFS

Donde TCPU es el tiempo que se le asigna a cada proceso, Tesp es el tiempo que tiene que esperar el proceso hasta ser ejecutado y Tpe es el tiempo promedio de espera el cual se obtiene de los datos anteriores.

n_proceso	TCPU	Tesp
1	8	13
2	4	5
3	3	1
4	1	0
Tpe:		4,75 ms

Tabla 2. Prueba de escritorio algoritmo SJF.

Aquí ya se encuentra la primera dificultad y es el que no se pudo implementar el tiempo de llegada para los dos siguientes algoritmos de planificación de procesos debido a la complicación que implica el implementar tiempos de llegada.

Finalmente se construyó un tabla la cuál se denominó el plan de prueba, que su función es la de identificar requisitos, riesgos, casos de prueba, entre otras cosas al momento de la planificación de un proyecto en general con los errores que tuvieron más impacto al momento del desarrollo de la codificación.

ITEMS	FUNCIONÓ		OBSERVACIONES/SOLUCION
	SI	NO	
Tiempos de llegada		X	Se presenta dificultad a la hora de desarrollar/implementarle tiempo de llegada al código, se complica bastante la ejecución del mismo, esto debido a que se desconoce la manera de realizarlo.
Valores incorrectos (char)	X		Al momento de ingresarle caracteres el programa muestra un mensaje para que el usuario ingrese un valor que sea correcto (número).
Tiempos de CPU	X		NO
Prioridad	X		NO
Quantum	X		NO
Diagrama de Gantt	X		NO
Tiempo promedio de espera	X		NO
Tiempo de espera	X		NO

Tabla 3. Plan de pruebas

5. Conclusiones

Se concluye que los algoritmos de planificación de procesos son muy útiles en los sistemas operativos ya que es debido a ellos que se conoce el concepto conocido como multiprogramación (pseudoparalelismo) y gracias a eso la implementación/ejecución de n procesos se hace más optimizado porque el procesador nunca se detiene hasta que termine los procesos. A pesar de la dificultad de implementar tiempos de llegada en la codificación se logró entender lo elemental de cada algoritmo visto durante la práctica.

Referencias.

[1]https://www.ecured.cu/Planificación_de_procesos_en_un_sistema_operativo

[2]https://www.lucidchart.com

[3]https://www.codeblocks.org