



Sistemas operativos

Algoritmo SJF

Alumno:

1.- Omar Santos Bernabé
osb0005@alu.ubu.es

Profesores:

José Manuel Sáiz

Enrique Sierra

Leticia Curiel

Alumnos 2014-2015:

ANTON GARCIA, ADRIÁN

BAILADOR PANERO, ADRIÁN

BASURTO, NUÑO

BENITO, JUAN FRANCISCO

BERLINCHES, ERIC

CASTRO HURTADO, VÍCTOR

GARCÍA GUTIÉRREZ, ÁLVARO

GORDO ORIVE, ZAIRA

MERINO, SHEILA

MILLÁN, MARCOS

SAIZ, FRANCISCO

ALEJANDRO CABALLERO GIL

RUBÉN URUÑUELA HERRERA

ADRIÁN MARCOS BATLLE,

IVÁN IGLESIAS CUESTA

ÁLVARO RUIFERNANDEZ PALACIOS

Alumnos 2015-2016:

OMAR SANTOS BERNABÉ

Índice de contenido

<u>1. Algoritmo con el que se trabaja y descripción.....</u>	<u>4</u>
<u>2. Ejercicios de ejemplo.....</u>	<u>5</u>
<u>2.1. Ejercicio hecho a mano:.....</u>	<u>5</u>
<u>Ejercicio1.....</u>	<u>5</u>
<u>Ejercicio2.....</u>	<u>6</u>
<u>2.2. Ejercicio hecho con el algoritmo.....</u>	<u>7</u>
<u>Ejercicio1.....</u>	<u>7</u>
<u>Ejercicio2.....</u>	<u>12</u>
<u>3. Datos de interes sobre código.....</u>	<u>18</u>

1. Algoritmo con el que se trabaja y descripción

SJF (Shortest Job First)

El algoritmo establece para la planificación una relación entre proceso y tiempo de la CPU. El algoritmo ejecuta en primer lugar los procesos de menor tamaño teniendo el caso base de que no haya procesos con misma duración, en ese lugar se realiza el algoritmo FCFS(First Come First Served) a esos dos procesos mediante el cual se ejecuta primero el proceso que antes llegue.

Este algoritmo presenta una gran ventaja, pues el tiempo de espera será mucho menor, porque mientras los procesos de tiempo inferior terminan y ocupan tiempo en operaciones de E/S, la CPU se ocupa de resolver el proceso con mayor tiempo, un algoritmo muy óptimo.

2. Ejercicios de ejemplo

2.1. Ejercicio hecho a mano:

Ejercicio1

Realizando el algoritmo de forma manual, con los siguientes datos, que son los mismos que introduciremos en la ejecución del algoritmo a ordenador, nos quedaría tal que así:

Procesos	Llegada	Ejecución
P1	0	3
P2	2	4
P3	2	2
P4	3	1

El resultado final de la ejecución sería el siguiente:

P1	P1	P1	P4	P3	P3	P2	P2	P2	P2
----	----	----	----	----	----	----	----	----	----

Este sería el correcto funcionamiento del algoritmo realizándolo de forma manual:

En primer lugar se ejecutaría el proceso p1 porque no hay ningún proceso en ejecución puesto que es el instante cero y no hay ningún otro proceso de menor tiempo ejecución que llegue en ese instante.

Seguidamente se ejecuta el proceso p4 ya que llega en el instante de tiempo tres, y a pesar de que el proceso p2 y p3 se encuentran en espera, se ejecuta antes este ya que tiene un tiempo de ejecución menor y se ejecuta con un tiempo en espera de cero.

A continuación se ejecuta el proceso p3, que a pesar de que el p2 llega en el mismo instante, como este tiene menor tiempo de ejecución es el primero en llevarse a cabo aunque tiene que estar dos quantum de tiempo en espera.

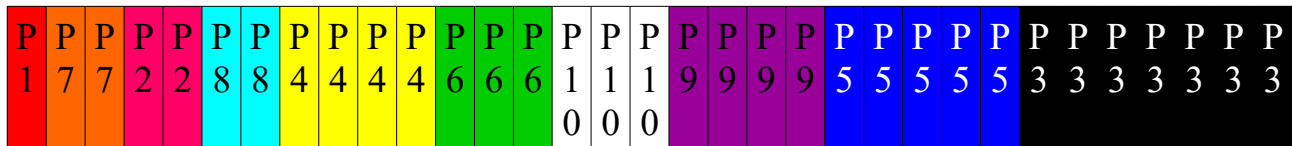
Finalmente se ejecuta el proceso p2, que a pesar de haber estado en espera desde el instante dos, por lo que tiene un tiempo de espera de cuatro quantum, es el proceso de los que se encontraban en espera con un mayor tiempo de ejecución, y por la definición del algoritmo, se ejecutan primero los procesos que menos tiempo de ejecución tengan.

Ejercicio2

Realizaremos otro ejemplo para demostrar cómo funciona la ejecución del algoritmo al introducir los datos desde un archivo o fichero. Estos son los datos que se encuentran en el fichero:

Procesos	Llegada	Ejecución
P1	0	1
P2	2	2
P3	4	7
P4	5	4
P5	6	5
P6	8	3
P7	1	2
P8	3	2
P9	15	4
P10	13	3

El resultado final de la ejecución con estos datos sería:



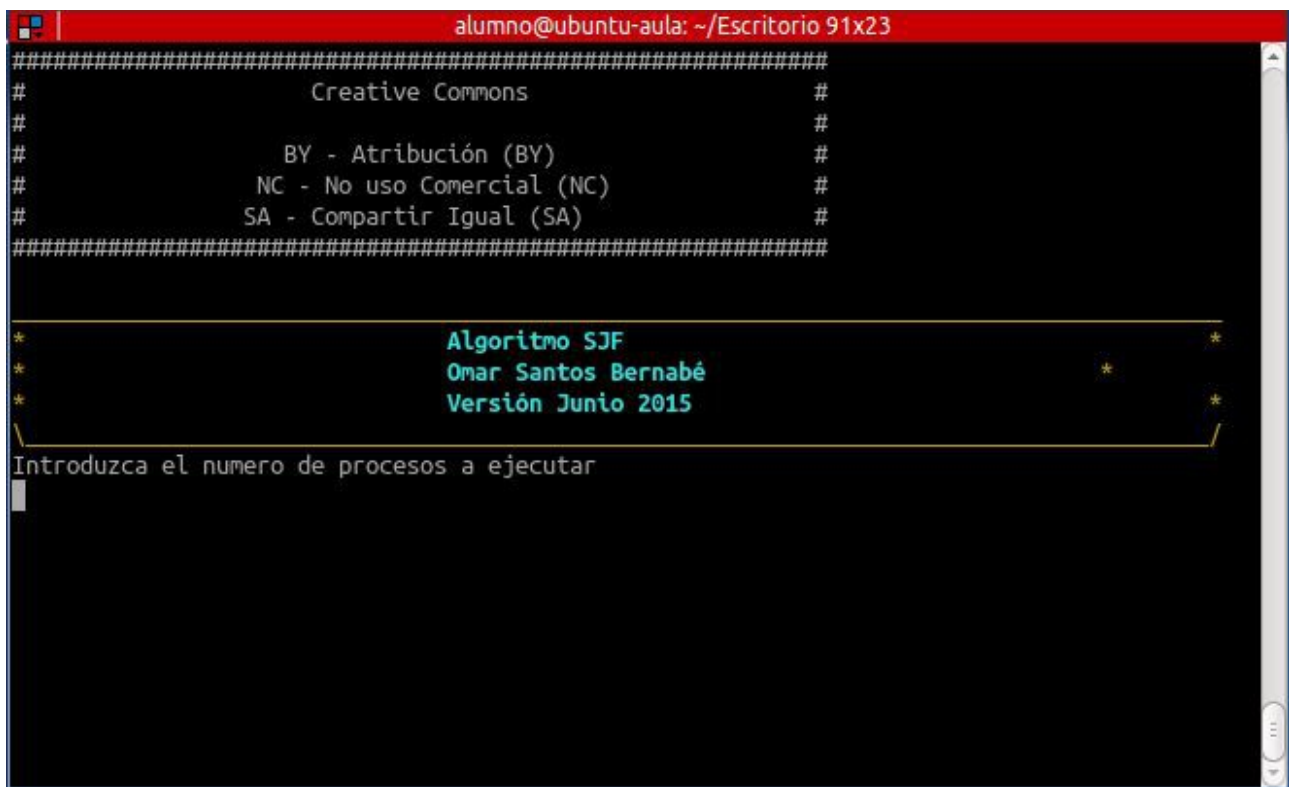
En primer lugar se ejecuta p1 ya que tiene tiempo de llegada 0 y no hay ningún otro proceso con llegada en ese instante. A continuación, en el instante 1 llega p7 y como es el único proceso que llega en ese instante se ejecuta. Durante la ejecución de p7 llega el proceso p2 y tiene que esperar un quantum de tiempo hasta poder ejecutarse en el instante de tiempo 3, que aunque en ese instante llega el proceso p8 como los dos tienen el mismo tiempo de ejecución se realiza el algoritmo FCFS entre ellos dos. En el instante 4 llega el proceso p3, pero por el mismo motivo que antes por el algoritmo FCFS se ejecuta p8 antes y p3 se encola. En el instante 5 llega el proceso p4 que queda encolado hasta la terminación de p8 y hace FCFS con el proceso p3 y el proceso p5 que llega en el instante 6 y entonces en el instante 7 se ejecuta p4. A continuación se realiza FCFS entre los procesos p6 que llega en el instante 8, p3 y p5 y como su ejecución es menor, pues entra en ejecución el proceso p6, del mismo modo se ejecutan los procesos p10 que llega en el instante 13 y el proceso p9 que llega en el instante 15 que tienen una ejecución menor que p3 y p5. Finalmente se ejecuta el proceso p5 por tener menor tiempo de ejecución que p3, y por último se ejecuta el proceso p3.

2.2. Ejercicio hecho con el algoritmo

Ejercicio1

A continuación veremos mediante volcados de pantalla la ejecución del algoritmo con los datos del primer ejercicio realizado a mano:

Lo primero de todo que se tiene que realizar es ir a la carpeta o directorio en el que se encuentre el algoritmo y una vez estemos ahí situados utilizaremos el comando “./SJF.sh” para ejecutarlo. Una vez hemos ejecutado el algoritmo lo primero que nos apareciera será esto.

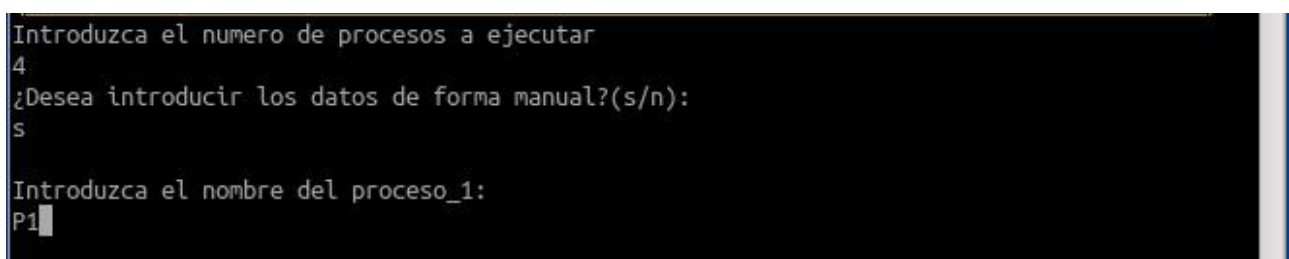


```
alumno@ubuntu-aula: ~/Escritorio 91x23
#####
#                Creative Commons                #
#                #                               #
#          BY - Atribución (BY)                  #
#          NC - No uso Comercial (NC)            #
#          SA - Compartir Igual (SA)             #
#                #                               #
#####

*                               *
*      Algoritmo SJF              *
*      Omar Santos Bernabé        *
*      Versión Junio 2015         *
*                               *
\                                 /

Introduzca el numero de procesos a ejecutar
█
```

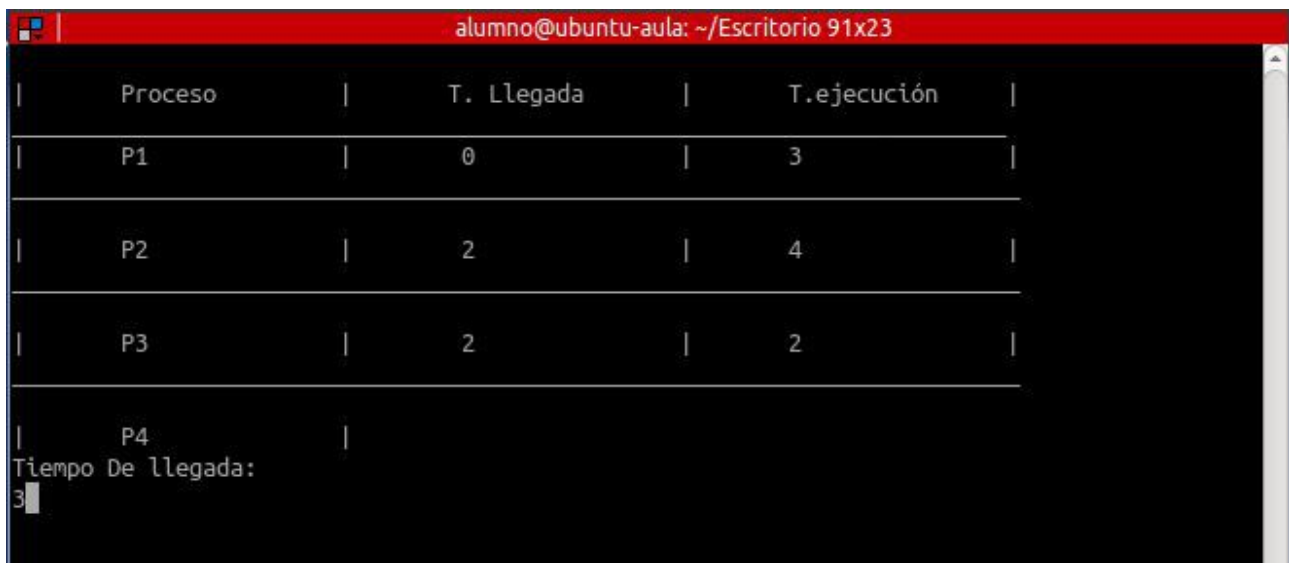
Aparece una licencia de creative commons que indica que es una versión nuestra y que no puede ser modificada sin autorización y que no puede usarse con un fin comercial. A continuación nos aparece la cabecera del algoritmo en la que podemos encontrar el nombre del algoritmo, el nombre del alumno que lo ha modificado por última vez y la versión a la que corresponde el algoritmo, en este caso la última versión realizada que es la de Junio del 2015.



```
Introduzca el numero de procesos a ejecutar
4
¿Desea introducir los datos de forma manual?(s/n):
s

Introduzca el nombre del proceso_1:
P1█
```

Como se podía ver en la imagen anterior al final nos pedia introducir el número de procesos que queremos ejecutar. Introducimos un total de cuatro procesos para que coincida con el número de procesos que hemos seleccionado en el primer ejercicio que hemor hecho de forma manual. A continuación nos pregunta si queremos introducir los datos de forma manual. En el caso de introducir como respuesta una s nos pedirá como se vera a continuación paso a paso cada uno de los datos necesarios para cada uno de los procesos. Si por el contrario introducimos una n, como se verá en el próximo ejemplo, cogerá los datos del fichero entradaSJF.txt en el que hay datos para varios procesos.



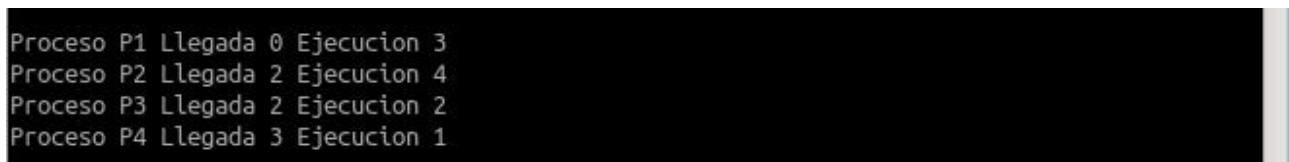
The image shows a terminal window with a red title bar that reads 'alumno@ubuntu-aula: ~/Escritorio 91x23'. Inside the terminal, a table is displayed with three columns: 'Proceso', 'T. Llegada', and 'T.ejecución'. The table contains four rows of data for processes P1, P2, P3, and P4. Below the table, the text 'Tiempo De llegada:' is followed by the number '3'.

Proceso	T. Llegada	T.ejecución
P1	0	3
P2	2	4
P3	2	2
P4		

Tiempo De llegada:
3

Lo primero que introducíamos como se ve en la imagen anterior era el nombre para el primer proceso y una vez introducido nos aparecia la siguiente tabla en la que podemos ver de forma mas clara los datos que ya han sido introducidos para así evatar introducir los mismos tiempos de llegada y de ejecución para distintos procesos, lo que sería inutil introducir varios procesos con los mismos datos puesto que de ese modo no conseguiríamos ver bien el funcionamiento de este algoritmo.

Como se puede observar se introducen los datos de forma ordenada cada uno para su procesos, y el algoritmo nos va pidiendo que introduzcamos primero el nombre que queremos para el proceso, a continuación el tiempo de llegada, el instante en el que dicho proceso llega para ejecutarse y finalmente el tiempo de ejecución que indica el tiempo que estará ejecutandose dicho proceso. Se realiza esto para cada uno de los procesos, en este caso para cuatro procesos que es el número de procesos que queríamos ejecutar y una vez hemos introducido los datos paraa todos los procesos el algoritmo se dispone a ejecutarse.



The image shows a terminal window with a black background. It displays a summary of the process data entered in the previous step, listing the process name, arrival time, and execution time for each of the four processes.

```
Proceso P1 Llegada 0 Ejecucion 3
Proceso P2 Llegada 2 Ejecucion 4
Proceso P3 Llegada 2 Ejecucion 2
Proceso P4 Llegada 3 Ejecucion 1
```


Lo primero que aparece una vez hemos terminado de introducir los datos de todos los procesos es una pequeña lista en la que podemos encontrarnos el nombre del proceso, el tiempo de llegada y el tiempo de ejecución de cada uno de los procesos que hemos introducido antes. De seguido podemos ver el orden de ejecución de los diferentes procesos en una pequeña tabla que contiene los procesos, su tiempo de llegada, su tiempo de ejecución, su tiempo de espera y su tiempo de respuesta.

	Proceso		Llegada		Ejecución		Espera		Respuesta	
*	P1	*	0	*	3	*	0	*	3	*
*	P4	*	3	*	1	*	0	*	1	*
*	P3	*	2	*	2	*	2	*	4	*
*	P2	*	2	*	4	*	4	*	8	*

* T.espera medio: 1 - * T.retorno medio: 3

Como podemos observar el primer proceso en ejecutarse es el proceso P1 que llega en el instante cero y como ningún otro proceso llega en ese instante pues comienza su ejecución, por lo que tiene cero unidades de tiempo de espera y como su ejecución es de tres unidades de tiempo pues su tiempo de respuesta es de tres unidades de tiempo. En el instante dos llegan los procesos P2 y P3 que se encolan puesto que se está ejecutando el proceso P1 y todavía no ha terminado de ejecutarse, y en el instante tres llega el proceso P4. En el instante tres el proceso P1 ha terminado su ejecución por lo que otro proceso puede entrar a ejecutarse, por lo que se realiza el algoritmo SJF (Shortest Job First) para saber cual de los siguientes procesos en cola debe ejecutarse. Según la definición del algoritmo el proceso que debe ejecutarse de los que están en espera es el proceso que menor tiempo de ejecución tiene, en este caso es el proceso P4 a pesar de haber sido el último en llegar a la cola. Entra a ejecutarse el proceso P4 y se ejecuta durante una unidad de tiempo y como había llegado en el instante tres y justo al entrar a ejecución su tiempo de espera será de cero, y por consecuencia su tiempo de respuesta será de uno puesto que es su tiempo de espera más su tiempo de ejecución. El siguiente proceso en ejecutarse es el proceso P3 por el mismo motivo que antes. Los dos procesos se encontraban en espera y habían llegado en el mismo instante pero el proceso P3 tiene menor tiempo de ejecución que el proceso P2 por lo tanto entra a ejecutarse antes, en el instante cuatro, por lo que su tiempo de espera es de dos unidades de tiempo porque había llegado en el instante dos y no entra a ejecución hasta el instante cuatro, y por eso su tiempo de respuesta es cuatro, porque a las dos unidades de tiempo se le suman las dos unidades de ejecución. Finalmente se ejecuta el proceso P2 que tiene un tiempo de ejecución de cuatro unidades de tiempo, su tiempo de espera es de cuatro unidades de tiempo también porque llega en el instante dos pero no empieza a ejecutarse hasta el instante seis, y su tiempo de respuesta es de ocho porque a su tiempo de ejecución (cuatro) hay que sumarle su tiempo de espera (cuatro).

¿Quieres abrir el informe? ([s],n): s

Por último se nos da la opción de abrir o no un informe con el resultado de la ejecución del algoritmo. A la hora de abrir el informe nos da dos opciones, introducir una s para abrirlo, que de igual forma funciona introduciendo una S o simplemente pulsando intro con el hueco vacío, o una n o N para no abrirlo y terminar la ejecución del algoritmo. En el caso de abrir el informe se nos abrirá un archivo txt como el siguiente.

```

informeSJF.txt
#####
#                               #
#          Creative Commons      #
#                               #
#          BY - Atribución (BY)  #
#          NC - No uso Comercial (NC) #
#          SA - Compartir Igual (SA) #
#                               #
#####

#####
#                               #
#          INFORME DE PRÁCTICA   #
#          GESTIÓN DE PROCESOS  #
#          -----              #
#          Nuevos alumnos:      #
#          Alumnos: Omar Santos Bernabé #
#          Sistemas Operativos 2º Semestre #
#          Grado en ingeniería informática (2015-2016) #
#                               #
#####

Proceso P1 Llegada 0 Ejecucion 3
Proceso P2 Llegada 2 Ejecucion 4
Proceso P3 Llegada 2 Ejecucion 2
Proceso P4 Llegada 3 Ejecucion 1

Proceso    Llegada    Ejecución    Espera    Respuesta
*   P1   *    0   *    3   *    0   *    3   *
*   P4   *    3   *    1   *    0   *    1   *
*   P3   *    2   *    2   *    2   *    4   *
*   P2   *    2   *    4   *    4   *    8   *

* T.espera medio: 1 - * T.retorno medio: 3

```

En el informe se puede ver de nuevo al principio la licencia del creative commons que como he explicado antes nos indica que el algoritmo no es de uso comercial y que no se debe modificar, sino que debe entregarse como se a recibido.

A continuación se observa una cabecera que indica el nombre del último alumno que ha realizado esta versión del algoritmo y el año o curso en el cual se ha realizado dicha versión.

Podemos observar los datos que se han introducido para la realización de este ejercicio en el orden que han sido introducidos. Los datos de muestran de la siguiente forma:

Proceso P1¹ Llegada 0² Ejecución 3³

¹.-Es el nombre que hemos dado al proceso.

².-Es el instante de tiempo en el que llega el proceso.

³.- Es el tiempo que tarda en ejecutarse dicho proceso.

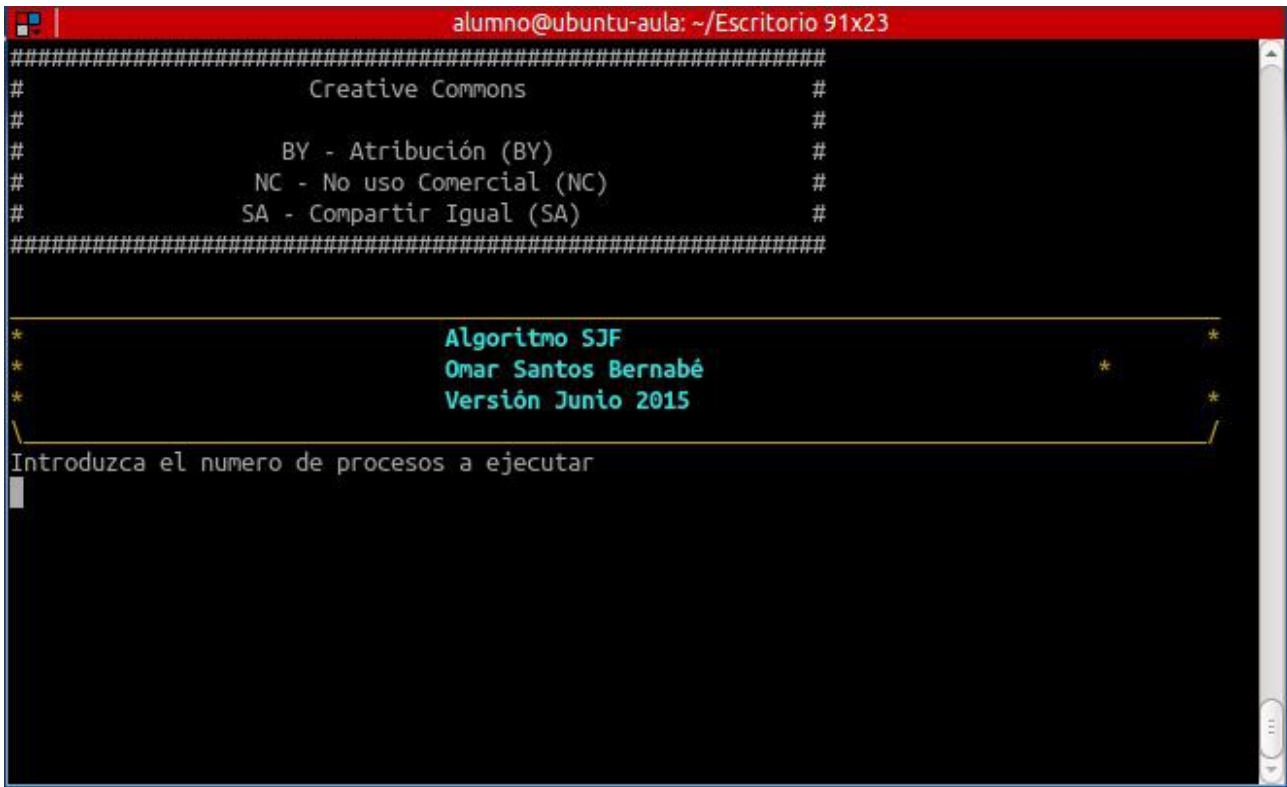
Finalmente observamos los resultados de la ejecución del algoritmo en la cual aparecen los procesos ordenados según su salida de ejecución junto con los datos del tiempo de llegada y de ejecución, y con el tiempo de llegada y el tiempo de respuesta de cada uno de los procesos. Al final aparecen los tiempos de espera medio y de retorno medio del algoritmo a partir de los datos que hemos introducido.

Una vez cerramos el fichero de texto se termina la ejecución del algoritmo.

Ejercicio2

A continuación veremos mediante volcados de pantalla la ejecución del algoritmo con los datos del primer ejercicio realizado a mano:

Lo primero de todo que se tiene que realizar es ir a la carpeta o directorio en el que se encuentre el algoritmo y una vez estemos ahí situados utilizaremos el comando “./SJF.sh” para ejecutarlo. Una vez hemos ejecutado el algoritmo lo primero que nos apareciera será esto.

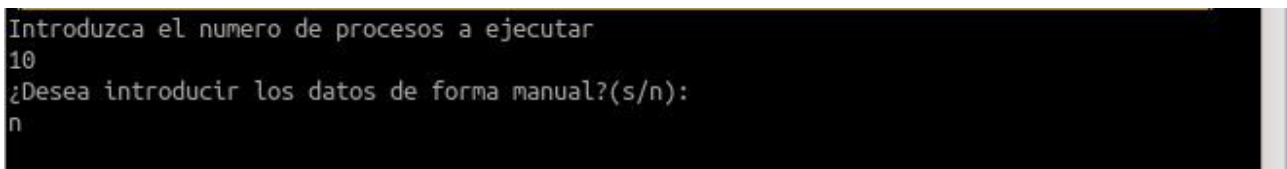


```
alumno@ubuntu-aula: ~/Escritorio 91x23
#####
#           Creative Commons           #
#                                     #
#           BY - Atribución (BY)       #
#           NC - No uso Comercial (NC) #
#           SA - Compartir Igual (SA)  #
#####

*                               *
*           Algoritmo SJF           *
*           Omar Santos Bernabé      *
*           Versión Junio 2015       *
*                               *
\_________________________________/

Introduzca el numero de procesos a ejecutar
|
```

Aparece una licencia de creative commons que indica que es una versión nuestra y que no puede ser modificada sin autorización y que no puede usarse con un fin comercial. A continuación nos aparece la cabecera del algoritmo en la que podemos encontrar el nombre del algoritmo, el nombre del alumno que lo ha modificado por última vez y la versión a la que corresponde el algoritmo, en este caso la última versión realizada que es la de Junio del 2015.



```
Introduzca el numero de procesos a ejecutar
10
¿Desea introducir los datos de forma manual?(s/n):
n
```

Como se podía ver en la imagen anterior al final nos pedía introducir el número de procesos que queremos ejecutar. Introducimos un total de 10 procesos para que coincidiera con el número de procesos que hemos seleccionado en el segundo ejercicio

que hemos hecho de forma manual. A continuación nos pregunta si queremos introducir los datos de forma manual. Introducimos una n para indicar que queremos obtener los datos del fichero de entradaSJF.txt en el que hay datos para un alto número de procesos.

```
Introduzca el nombre del proceso_1:
Tiempo De llegada:
Tiempo De Ejecución_1
Introduzca el nombre del proceso_2:
Tiempo De llegada:
Tiempo De Ejecución_2
Introduzca el nombre del proceso_3:
Tiempo De llegada:
Tiempo De Ejecución_3
Introduzca el nombre del proceso_4:
Tiempo De llegada:
Tiempo De Ejecución_4
Introduzca el nombre del proceso_5:
Tiempo De llegada:
Tiempo De Ejecución_5
Introduzca el nombre del proceso_6:
Tiempo De llegada:
Tiempo De Ejecución_6
Introduzca el nombre del proceso_7:
Tiempo De llegada:
Tiempo De Ejecución_7
Introduzca el nombre del proceso_8:
Tiempo De llegada:
Tiempo De Ejecución_8
Introduzca el nombre del proceso_9:
Tiempo De llegada:
Tiempo De Ejecución_9
Introduzca el nombre del proceso_10:
Tiempo De llegada:
Tiempo De Ejecución_10
█
```

Como se puede observar en la imagen el propio algoritmo va cogiendo en orden los datos que necesita para los diez procesos que vamos a ejecutar. Los datos se recogen en el mismo orden que se pedían de forma manual, primero el nombre que queremos dar al proceso, después el tiempo de llegada de dicho proceso y finalmente el tiempo de ejecución para ese proceso. Se realiza ese paso hasta recoger los datos para los diez procesos que queremos.

Como se puede observar se introducen los datos de forma ordenada cada uno para su procesos, y el algoritmo nos va pidiendo que introduzcamos primero el nombre que queremos para el proceso, a continuación el tiempo de llegada, el instante en el que dicho proceso llega para ejecutarse y finalmente el tiempo de ejecución que indica el tiempo que estará ejecutándose dicho proceso. Se realiza esto para cada uno de los procesos, en este caso para diez procesos que es el número de procesos que queríamos ejecutar y una vez ha recogido los datos del fichero para todos los procesos el algoritmo se dispone a ejecutarse.

```

Proceso P1 Llegada 0 Ejecucion 1
Proceso P2 Llegada 2 Ejecucion 2
Proceso P3 Llegada 4 Ejecucion 7
Proceso P4 Llegada 5 Ejecucion 4
Proceso P5 Llegada 6 Ejecucion 5
Proceso P6 Llegada 8 Ejecucion 3
Proceso P7 Llegada 1 Ejecucion 2
Proceso P8 Llegada 3 Ejecucion 2
Proceso P9 Llegada 15 Ejecucion 4
Proceso P10 Llegada 13 Ejecucion 3

```

Lo primero que aparece una vez hemos terminado de introducir los datos de todos los procesos es una pequeña lista en la que podemos encontrarnos el nombre del proceso, el tiempo de llegada y el tiempo de ejecución de cada uno de los procesos que hemos introducido antes. De seguido podemos ver el orden de ejecución de los diferentes procesos en una pequeña tabla que contiene los procesos, su tiempo de llegada, su tiempo de ejecución, su tiempo de espera y su tiempo de respuesta.

Proceso	Llegada	Ejecución	Espera	Respuesta
* P1 *	* 0 *	* 1 *	* 0 *	* 1 *
* P7 *	* 1 *	* 2 *	* 0 *	* 2 *
* P2 *	* 2 *	* 2 *	* 1 *	* 3 *
* P8 *	* 3 *	* 2 *	* 2 *	* 4 *
* P4 *	* 5 *	* 4 *	* 2 *	* 6 *
* P6 *	* 8 *	* 3 *	* 3 *	* 6 *
* P10 *	* 13 *	* 3 *	* 1 *	* 4 *
* P9 *	* 15 *	* 4 *	* 2 *	* 6 *
* P5 *	* 6 *	* 5 *	* 15 *	* 20 *
* P3 *	* 4 *	* 7 *	* 22 *	* 29 *

* T.espera medio: 4 - * T.retorno medio: 8

Como podemos observar el primer proceso en ejecutarse es el proceso P1 que llega en el instante cero y como ningún otro proceso llega en ese instante pues comienza su ejecución, por lo que tiene cero unidades de tiempo de espera y como su ejecución es de una unidad de tiempo pues su tiempo de respuesta es de una unidades. En el instante uno llega el proceso P7 que entra a ejecutarse porque el proceso P1 ya ha terminado su ejecución y no hay ningún otro proceso para comparar sus tiempos de ejecución. En el instante dos llega el proceso P2 y en el instante tres llega el proceso P8 que se encolan puesto que se esta ejecutando el proceso P7 y todavía no a terminado de ejecutarse. Como los dos procesos en cola tienen el mismo tiempo de ejecución el primero en ejecutarse según termina el proceso P7 es el proceso P2 que es el primero en llegar con una unidad de tiempo de espera porque llega en el instante dos pero no comienza a ejecutarse hasta el instante tres. Durante la ejecución del proceso P2 llega el proceso P3 en el instante cuatro, el proceso P4 en el instante cinco y el proceso P5 en el instante seis y se encolan. Una vez termina de ejecutarse el proceso P2 empieza a ejecutarse el proceso P8 porque de los que están en espera es el que menor tiempo de ejecución tiene. Se ejecuta durante dos unidades de tiempo y

como tiene dos unidades de tiempo de espera el tiempo de respuesta es de cuatro. A continuación entra a ejecutarse el proceso P4 porque llegaba en el instante cinco pero como su tiempo de ejecución es menor que el del proceso P3 y el proceso P5 pues comienza su ejecución y como entra a ejecutarse en el instante siete tiene dos unidades de tiempo de espera y su tiempo de respuesta final es de seis unidades de tiempo. Durante la ejecución del proceso P4 llega el proceso P6 que se encola, pero una vez termina la ejecución del proceso P4 comienza a ejecutarse porque de los tres procesos que habría en cola es el que menor tiempo de ejecución tiene. Su tiempo de espera es de tres unidades de tiempo porque desde que llega en el instante ocho no empieza su ejecución hasta el instante once y su tiempo de respuesta es el tiempo de espera de tres unidades más el tiempo de ejecución con un total de seis unidades. Durante su ejecución llega el proceso P10 y se encola pero como de los dos procesos que hay en cola es el que menos tiempo de ejecución tiene por lo que según termina la ejecución del proceso P6 comienza a ejecutarse con un tiempo de espera de una unidad de tiempo y un tiempo de ejecución de tres unidades de tiempo, por lo que su tiempo de respuesta total es de cuatro unidades. Durante la ejecución del proceso P10 llega el proceso P9 que se encola, y una vez termina el proceso P10 empieza la ejecución del proceso P9, una vez termina se ejecuta el proceso P5 que tiene menor tiempo de ejecución que el proceso P3 y finalmente el proceso P3 es el que se ejecuta.



```
¿Quieres abrir el informe? ([s],n): s
```

Por último se nos da la opción de abrir o no un informe con el resultado de la ejecución del algoritmo. A la hora de abrir el informe nos da dos opciones, introducir una s para abrirlo, que de igual forma funciona introduciendo una S o simplemente pulsando intro con el hueco vacío, o una n o N para no abrirlo y terminar la ejecución del algoritmo. En el caso de abrir el informe se nos abrirá un archivo txt como el siguiente.

	Proceso		Llegada		Ejecución		Espera		Respuesta	
*	P1	*	0	*	1	*	0	*	1	*
*	P7	*	1	*	2	*	0	*	2	*
*	P2	*	2	*	2	*	1	*	3	*
*	P8	*	3	*	2	*	2	*	4	*
*	P4	*	5	*	4	*	2	*	6	*
*	P6	*	8	*	3	*	3	*	6	*
*	P10	*	13	*	3	*	1	*	4	*
*	P9	*	15	*	4	*	2	*	6	*
*	P5	*	6	*	5	*	15	*	20	*
*	P3	*	4	*	7	*	22	*	29	*

* T.espera medio: 4 - * T.retorno medio: 8

Finalmente observamos los resultados de la ejecución del algoritmo en la cual aparecen los procesos ordenados según su salida de ejecución junto con los datos del tiempo de llegada y de ejecución, y con el tiempo de llegada y el tiempo de respuesta de cada uno de los procesos. Al final aparecen los tiempos de espera medio y de retorno medio del algoritmo a partir de los datos que hemos introducido.

Una vez cerramos el fichero de texto se termina la ejecución del algoritmo.

3. Datos de interes sobre código

```
echo "Introduzca el nombre del proceso $p:"
nombre=`cat entradaSJF.txt | cut -d"*" -f$pp | cut -d" " -f3`
Comprobarn $nombre #envio nombre a comprobar por la funcion
proceso[$p]=$nombre; #añado a el vector ese nombre

echo "Tiempo De llegada:"
llegad=`cat entradaSJF.txt | cut -d"*" -f$pp | cut -d" " -f5`
#echo >> informeSJF.txt

llegada[$p]=$llegad; #añado al vector ese numero

echo "Tiempo De Ejecución $p"
tiemp=`cat entradaSJF.txt | cut -d"*" -f$pp | cut -d" " -f7`
#echo ejecucion $tiemp >> informeSJF.txt

tiempo[$p]=$tiemp; #añado al vector ese numero
```

En esta imagen podemos ver la forma que tiene el algoritmo de coger los datos del fichero entradaSJF.txt cuando seleccionamos en la ejecución que no queremos introducir los datos de forma manual. Lo que realiza para obtener el nombre es que abre el fichero entradaSJF.txt y de ahí, cada línea después del asterisco es un nuevo proceso, por lo que coge para el nombre lo que hay después del asterisco entre el segundo y tercer espacio. Del mismo modo coge los datos correspondientes al tiempo de llegada que se encuentra justo antes del quinto espacio y finalmente el tiempo de ejecución que está justo antes del séptimo espacio. Este procedimiento se hace invisible al usuario y se realiza para coger los datos para los diferentes procesos que se desean introducir.

```
#promedios
echo "" >> ejemplo.txt
echo "" >> informeSJF.txt
echo "* T.espera medio: $promedio_espera - * T.retorno medio: $promedio_respuesta " >> informeSJF.txt
echo "" >> informeSJF.txt
echo -e "\e[0;31m * T.espera medio: $promedio_espera - * T.retorno medio: $promedio_respuesta \e[0m" >> ejemplo.txt
echo "" >> ejemplo.txt

cat ejemplo.txt
```

Un detalle que cabe destacar del código pueden ser las líneas repetidas. Como se puede ver en la imagen hay dos líneas repetidas, una que se guarda en informeSJF.txt y otra que se guarda en ejemplo.txt. La que se guarda en informeSJF.txt no contiene colores ya que los caracteres para pintar las líneas en la consola, el editor de texto no los reconoce y entonces nos guarda la línea con el carácter en vez de la línea coloreada, por eso se guardan en dos ficheros diferentes.

Como en el fichero ejemplo.txt guardamos la salida del algoritmo según se va ejecutando, justo antes de terminar el programa colocamos un cat ejemplo.txt para que así pueda verse la salida del algoritmo en su ejecución en la consola.