

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

Departamento de ciencias computacionales.

Ingeniería en computación



Seminario de Solución de Problemas de Sistemas Operativos - D01.

Violeta del Rocío Becerra Velázquez

“Programa 5. Algoritmo de planificación
RR(Round-Robin)”

2023A

22/03/2023

Flores Estrada Abraham Miguel Angel

Olguín Hernández Jair Benjamín

Índice.

Índice.	2
Objetivo.	3
Lenguaje utilizado.	3
Desarrollo del programa.	3
Conclusiones.	7

Objetivo.

El algoritmo de planificación Round Robin es uno de los algoritmos más populares utilizados en los sistemas operativos para la gestión de procesos. Este algoritmo se basa en la idea de asignar a cada proceso una cantidad fija de tiempo de CPU, llamado "quantum", y luego rotar los procesos en ejecución en un orden circular. Es decir que establece un reloj que asigna una cantidad de tiempo igual a cada proceso para ejecutarse en secuencia. Cuando el tiempo asignado a un proceso ha pasado, se interrumpe y se guarda su estado actual, y se da paso al siguiente proceso en la cola. El proceso interrumpido se mueve al final de la cola y espera su próximo turno. Por lo que buscamos crear un programa que logre simular el algoritmo RR, donde podamos aprovechar el programa anterior pero ahora aplicando un algoritmo diferente, haciendo uso de las funciones principales pero donde se tiene que agregar el quantum y que el programa se adapte para poder funcionar con este.

Lenguaje utilizado.

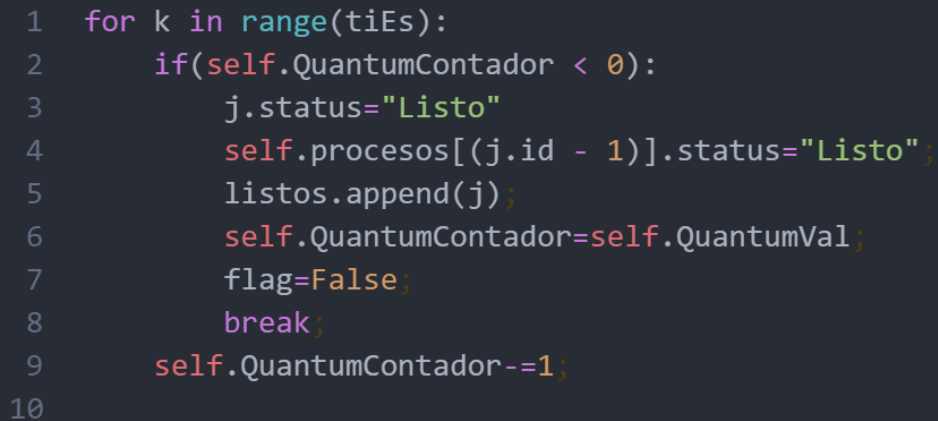
Continuamos con el uso de python, debido a que esta actividad se va a desarrollar sobre el código pasado, donde solo tendremos que agregar las funcionalidades pendientes como el quantum y que los procesos cambian cada cierto tiempo.

Desarrollo del programa.

En este caso debido a los requerimientos del programa ambas interfaces tuvieron cambios pero estos los veremos más detalladamente adelante, pero recordemos que ahora tenemos una función que nos ayuda a crear los procesos no solamente al inicio sino al presionar la tecla "n", donde estos son guardados en una cola de listos que solo permite tener 4 en memoria y podemos ver el tiempo de los procesos en cualquier momento.

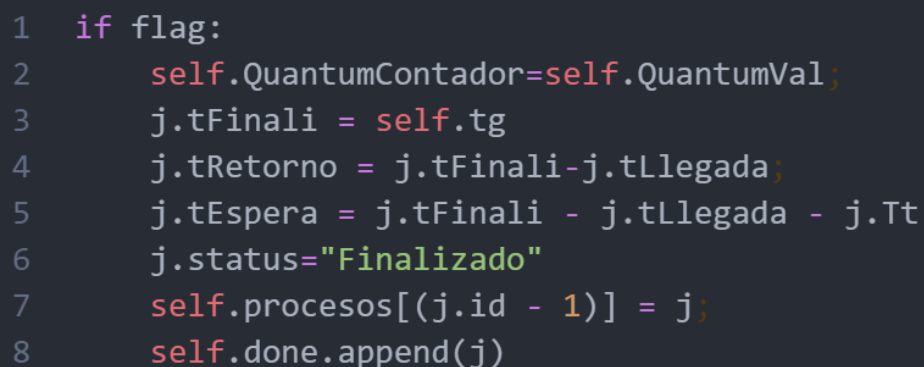
Por lo que los cambios que debemos hacer están en las iteraciones que hacemos sobre cada proceso, donde guardamos el valor del quantum en una variable que esta irá disminuyendo hasta alcanzar el valor de 0, una vez que este contador alcance este valor lo que se hará es cambiar el status del proceso nuevamente a listo, donde le tenemos que restar al id uno para ajustar el índice de la lista y volveremos agregar este proceso a la cola de listos. Así mismo como el contador de quantum llegó a cero volvemos a igual el valor de este al valor que se declaró al

inicio. Y tenemos una bandera la cual nos ayuda a saber cuando el proceso ha sido terminado.



```
1  for k in range(tiEs):
2      if(self.QuantumContador < 0):
3          j.status="Listo"
4          self.procesos[(j.id - 1)].status="Listo";
5          listos.append(j);
6          self.QuantumContador=self.QuantumVal;
7          flag=False;
8          break;
9      self.QuantumContador-=1;
10
```

Donde esto hará que cada vez que un proceso esté en ejecución solo pase el valor del quantum y se agregue nuevamente a la cola de listos. Donde una vez que el proceso haya terminado completamente, se vuelve asignar el valor del quantum al inicial, ya que pudo haber quedado con algun otro numero; asi mismo asignamos los valores de los distintos tiempos y agregamos el proceso a terminados.



```
1  if flag:
2      self.QuantumContador=self.QuantumVal;
3      j.tFinali = self.tg
4      j.tRetorno = j.tFinali-j.tLlegada;
5      j.tEspera = j.tFinali - j.tLlegada - j.Tt
6      j.status="Finalizado"
7      self.procesos[(j.id - 1)] = j;
8      self.done.append(j)
```

Veamos el programa en ejecución. Donde lo primero que vemos es que nos pide los números de procesos y así mismo el valor que le queremos asignar al quantum. En nuestro caso pondremos 8 procesos y el valor de 6 al quantum.

MainWindow

Simulador de procesos

Num de procesos:

Quantum:

Aceptar

Después de esto al iniciar el programa veremos la cantidad de nuevos procesos pendientes, el valor que se le asignó al quantum, la cola de listos y el proceso en ejecución, así mismo con el avance del programa se llenará la tabla de procesos terminados.

MainWindow

Simulador de procesos

Nuevos procesos:

Quantum:

Listos

	Id	TiEs	TT
1	2	8	0
2	3	13	0
3	4	10	0

Contador Global:

Proceso en Ejecución

	1
Id	1
Operacion	472 * 347
Tiempo	11
Tiempo Total	1
Tiempo Restante	10
Quantum	5

Iniciar

Proceso bloqueados

ID	TB
----	----

Procesos Terminados

Id	Op	Res
----	----	-----

Ver tiempos

Donde si dejamos que el programa avance un poco podemos ver que el tiempo total de los procesos en la cola de listos tienen un valor de 6, ya que este es el valor del quantum.



Simulador de procesos

Nuevos procesos: **4**

Quantum: **6**

Listos

	Id	TiEs	TT
1	1	11	6
2	2	8	6
3	3	13	6

Proceso en Ejecución

	1
Id	4
Operacion	148 / 722
Tiempo	10
Tiempo Total	1
Tiempo Restante	9
Quantum	5

Proceso bloqueados

ID	TB
----	----

Procesos Terminados

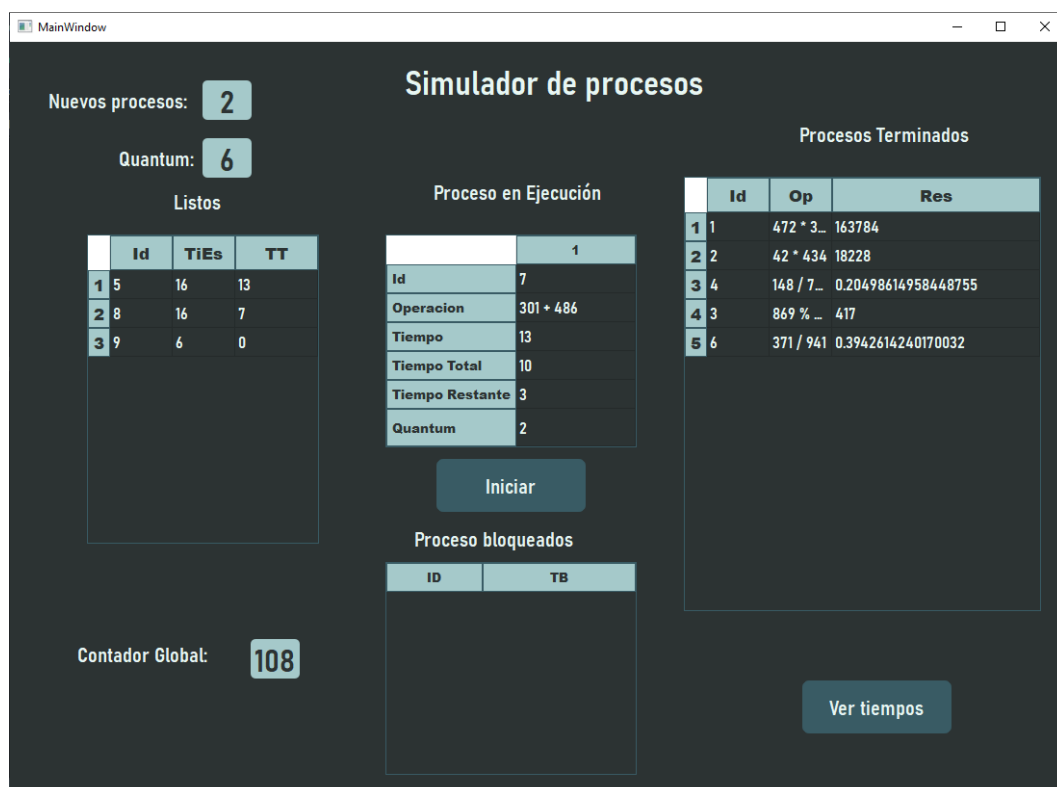
Id	Op	Res
----	----	-----

Contador Global: **19**

Iniciar

Ver tiempos

De igual manera podemos observar que pueden haber procesos que terminen antes que otros, como en nuestro caso que tenemos el proceso de ID 4,3 y después 6, debido a que estos pudieron tener un menor tiempo de espera y por los ciclos realizados pudieron terminar antes o después.



Simulador de procesos

Nuevos procesos: **2**

Quantum: **6**

Listos

	Id	TiEs	TT
1	5	16	13
2	8	16	7
3	9	6	0

Proceso en Ejecución

	1
Id	7
Operacion	301 + 486
Tiempo	13
Tiempo Total	10
Tiempo Restante	3
Quantum	2

Procesos Terminados

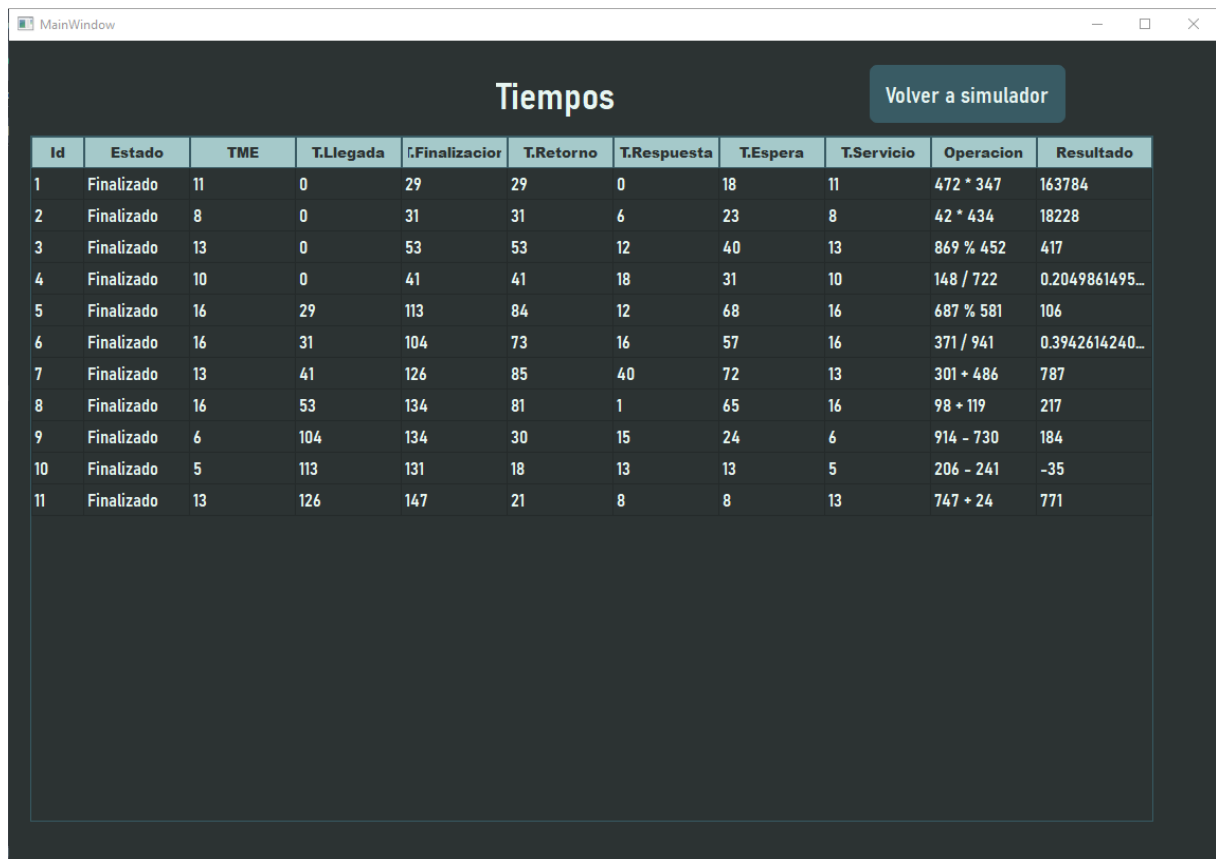
	Id	Op	Res
1	1	472 * 3...	163784
2	2	42 * 434	18228
3	4	148 / 7...	0.20498614958448755
4	3	869 % ...	417
5	6	371 / 941	0.3942614240170032

Contador Global: **108**

Iniciar

Ver tiempos

Así mismo se puede ver que aún se pueden agregar procesos presionando la tecla “n”, donde pasamos de tener 8 procesos iniciales, a tener 11, donde son 4 en ejecución, 5 terminados y 2 pendientes. Y una vez finalizado todos los procesos se nos muestra la tabla de tiempos de los procesos.



Id	Estado	TME	T.Llegada	T.Finalizaci3n	T.Retorno	T.Respuesta	T.Espera	T.Servicio	Operaci3n	Resultado
1	Finalizado	11	0	29	29	0	18	11	472 * 347	163784
2	Finalizado	8	0	31	31	6	23	8	42 * 434	18228
3	Finalizado	13	0	53	53	12	40	13	869 % 452	417
4	Finalizado	10	0	41	41	18	31	10	148 / 722	0.2049861495...
5	Finalizado	16	29	113	84	12	68	16	687 % 581	106
6	Finalizado	16	31	104	73	16	57	16	371 / 941	0.3942614240...
7	Finalizado	13	41	126	85	40	72	13	301 + 486	787
8	Finalizado	16	53	134	81	1	65	16	98 + 119	217
9	Finalizado	6	104	134	30	15	24	6	914 - 730	184
10	Finalizado	5	113	131	18	13	13	5	206 - 241	-35
11	Finalizado	13	126	147	21	8	8	13	747 + 24	771

Donde aqu  podemos apreciar m s el caso de los procesos anteriormente mencionados, que el proceso de ID 4 ten a un tiempo esperado menor al 3 por lo que este finaliz  antes. Y en el caso del proceso 5 y 6 ten an un tiempo esperado igual, pero el proceso 5 tuvo una interrupci n por lo que el proceso 6 termin  antes.

Conclusiones.

El desarrollo de este programa fue m s sencillo de lo que pens bamos, ya que al inicio desconoc amos sobre totalmente sobre el algoritmo round robin por lo que pens bamos que el adaptar el programa que ten amos a este algoritmo iban a requerir hacer muchos cambios, sin embargo fue algo sencillo donde solo tuvimos que validar algunas cosas para evitar errores y tener cuidado con los tiempos del quantum. Y viendo la simulaci n del programa se logra entender mucho mejor el c mo es que funciona, y qu  ventajas otorgan los quantums, donde en el ejemplo

mostrado no se vieron tantos casos debido a que el quantum tiene un valor algo pequeño, pero en casos mucho más grandes se puede lograr ver que los procesos de menor tiempo serán los primeros en terminar, y que así mismo todos los procesos se irán trabajando de poco en poco. Aunque aún así el que tengan un menor tiempo esperado no significa que sean más o menos importantes, y aún no existe alguna prioridad que puedan definir la importancia de algún proceso.