

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 4. Algoritmo de planificación FCFS
continuación.”

2023A

10/03/2023

Flores Estrada Abraham Miguel Angel

Olguín Hernández Jair Benjamín

Índice.

Índice.	2
Objetivo.	3
Lenguaje utilizado.	3

Objetivo.

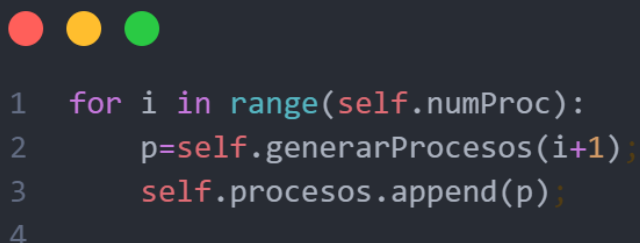
Recordemos que el algoritmo de planificación FCFS (First Come, First Served) es uno de los métodos de planificación de procesos en un sistema operativo. Donde asigna la CPU al primer proceso que llega y lo ejecuta hasta que completa su tiempo de ejecución, momento en el cual se pasa al siguiente proceso en la cola. Donde en la actividad anterior pudimos simular de manera sencilla la primera parte del algoritmo FCFS, haciendo que los procesos entren a una cola de listos y así mismo los procesos pueden llegar a tener alguna interrupción, y deberán pasar a lista de bloqueados donde tendrán que pasar un tiempo de espera. Por lo que buscamos continuar con el programa anterior pero ahora implementando las funciones de agregar nuevos procesos y la tabla de procesos. Es decir que puedas agregar nuevos procesos con solo presionar la tecla n y ver la tabla de todos los procesos que hay hasta el momento.

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.


Desarrollo del programa.

En este caso debido a los requerimientos del programa las interfaces no tuvieron ningún cambio, ya que los cambios fueron dentro de las funcionalidades y especificaciones, donde anteriormente se pedían los números de procesos y se crean directamente después de pedirlos de una manera lineal, ya que esta parte de código no se volvía a acceder nuevamente. Pero ahora lo que hacemos es tener nuestra función “generarProcesos”, que básicamente genera el proceso con la información en aleatorio, pero ahora esta función es accesible desde dos casos. El primero es cuando se ingresan los primeros “n” procesos a simular, donde hará un for que irá llamando la función anterior y agregando el proceso generado a nuestra lista de procesos.




```
1  for i in range(self.numProc):
2      p=self.generarProcesos(i+1);
3      self.procesos.append(p);
4
```

El segundo caso cuando llamamos a esta función es cuando se quiere crear un nuevo proceso mientras la simulación está en proceso. Donde tenemos nuestra bandera que verifica si la tecla "N" ha sido presionada, en dado caso que si esta llama a nuestra función donde se generan los procesos y la agrega a la misma lista de procesos. Donde tenemos que verificar que el número de procesos en memoria sea menor que 4, por lo que checamos el número de proceso actual (último proceso en agregarse a listos) y verificamos que los procesos en bloqueados y listos sean menores que 4. Por lo que si tenemos menos de 4 procesos en memoria el proceso que se agregue al presionar "N" tendrá que entrar a la cola de listos. Y volvemos a poner en falso nuestra bandera, para poder saber si nuevamente fue presionada la tecla "N".




```
1  if self.FNuevo:
2      p=self.generarProcesos(i)
3      self.procesos.append(p)
4      if(act<numProc) and (len(bloq)+len(listos) < 4):
5          self.procesos[act].tllegada=self.tg
6          listos.append(self.procesos[act])
7          act+=1
8      self.FNuevo=False
```

Por otra parte también agregamos nuestra bandera que revisa si la tecla "T" fue presionada, en caso de que se presionada se activa la bandera y llamamos a la función que se encarga de obtener todos los tiempos actuales de los procesos, así mismo ponemos en pausa el programa hasta que se presione la tecla "C" para continuar.




```
1  if self.FTabla:
2      self.actTimes();
3      self.FPausa=True;
4      if self.FPausa:
5          while True:
6              if self.FPausa == False:
7                  break;
8      self.FTabla=False
```

Donde anteriormente a esto ya mandamos los procesos a la cola de listos o sabemos si es un proceso nuevo. Lo que hacemos al momento de crear el proceso se genera con un estatus de "Nuevo", por lo que todos los procesos al iniciar están como nuevos y dependiendo el estado por el que pasen este se va cambiando.



```
1 class proceso(object):
2     def __init__(self,Id,operacion,res,tiEs):
3         self.id = Id
4         self.atendido=False
5         self.operacion = operacion
6         self.status="Nuevo"
```

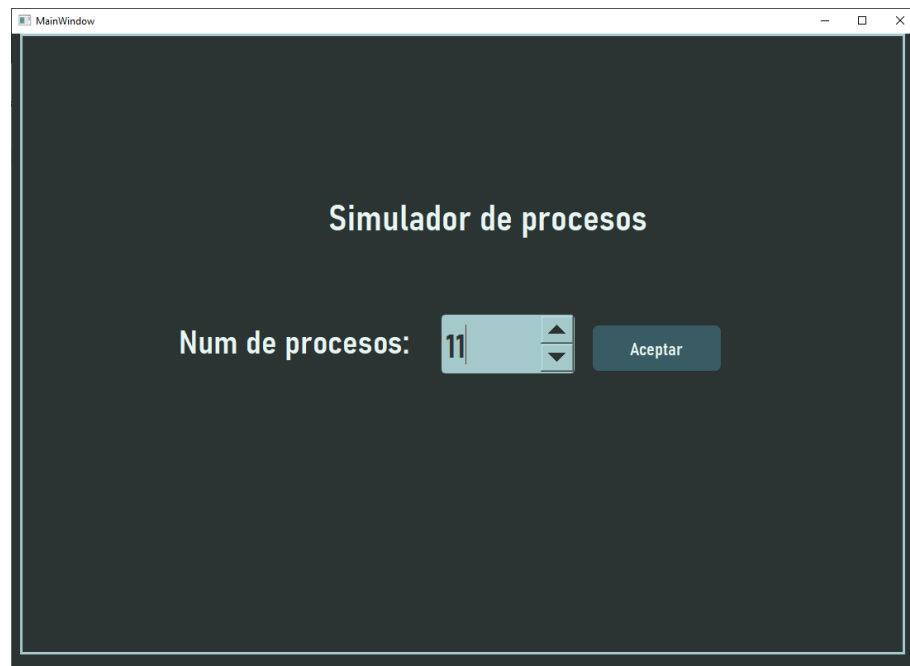
Después de esto en cuanto entren a nuestra cola de listos su estatus cambiará a "Listo", donde solo podemos tener 4 procesos con este estatus. Y en dado caso que algún o todos los procesos reciban una interrupción su estatus cambia a "Bloqueado". Y por último tenemos el caso que el proceso haya terminado, en este caso solo comparamos con una bandera que nos indica si el proceso ha terminado.



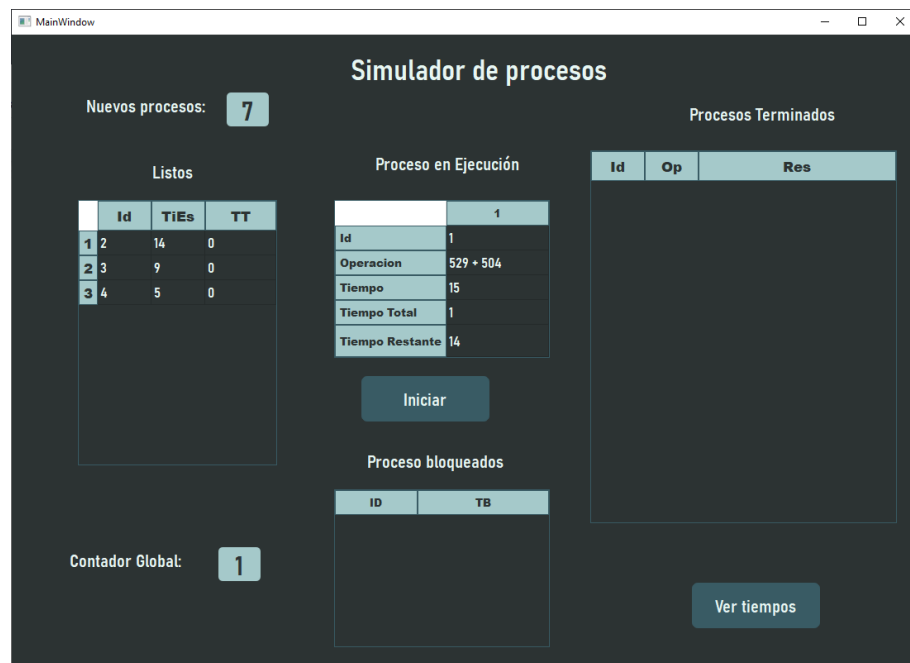
```
1 while (act)<numProc and i<4:
2     self.procesos[act].status="Listo"
3     listos.append(self.procesos[act])
4
5 if self.FInterrupcion:
6     j.Tb = 8
7     j.status="Bloqueado"
8     bloq.append(j)
9
10 if terminado:
11     j.status="Finalizado"
12     j.tFinali = self.tg
```

Veamos el programa en ejecución:

Inicia con pidiendonos el número de procesos:



Al iniciar se pondrán los 4 procesos en memoria, donde uno está directamente en ejecución y los 3 faltantes en la cola de listos. Así mismo se nos muestra que tenemos 7 procesos nuevos, donde este número ya puede tanto disminuir como aumentar.



En este caso dejamos que algunos procesos terminaran con algunas interrupciones para ver las tablas de tiempo.

MainWindow

Simulador de procesos

Nuevos procesos: **2**

Listos

	Id	TiEs	TT
1	6	10	3
2	7	16	1
3	8	6	1

Proceso en Ejecución

	1
Id	9
Operacion	470 + 706
Tiempo	12
Tiempo Total	4
Tiempo Restante	8

Iniciar

Proceso bloqueados

ID	TB
----	----

Procesos Terminados

	Id	Op	Res
1	1	529 + 5...	1033
2	2	854 * 9...	ERROR
3	3	300 % ...	91
4	4	285 / 4...	0.6834532374100719
5	5	43 - 873	-830

Ver tiempos

Contador Global: **92**

Donde podemos notar que los procesos del 1 al 5 tienen el estado de finalizado, cada uno de sus tiempos y su resultado. Los procesos del 6 al 8 están con el estado en listo, ya que estos se encuentran en la cola de listos, y el proceso 9 está en ejecución. Así mismo el proceso 10 y 11 están con el estatus de nuevo ya que solo han sido creados y el único tiempo que tienen es el tiempo esperado. También vemos que solo los procesos que han pasado por listos tienen su tiempo de llegada, respuesta, espera y servicio.

MainWindow

Tiempos

Volver a simulador

	Id	Estado	TME	T.Llegada	L.Finalizaci	T.Retorno	T.Respuesta	T.Espera	T.Servicio	Operacion	Resultado
1	1	Finalizado	15	0	55	55	0	40	15	529 + 504	1033
2	2	Finalizado	14	0	58	58	8	53	5	854 * 946	ERROR
3	3	Finalizado	9	0	64	64	9	55	9	300 % 209	91
4	4	Finalizado	5	0	75	75	10	70	5	285 / 417	0.683453237...
5	5	Finalizado	13	55	88	33	9	20	13	43 - 873	-830
6	6	Listo	10	58	NULL	NULL	17	32	3	517 * 614	NULL
7	7	Listo	16	64	NULL	NULL	14	28	1	239 + 583	NULL
8	8	Listo	6	75	NULL	NULL	4	17	1	721 + 422	NULL
9	9	Ejecucion	12	88	NULL	NULL	0	0	5	470 + 706	NULL
10	10	Nuevo	16	NULL	NULL	NULL	NULL	NULL	NULL	666 / 910	NULL
11	11	Nuevo	14	NULL	NULL	NULL	NULL	NULL	NULL	353 - 113	NULL

En este caso presionamos la tecla “N” para agregar algunos procesos nuevos, pasamos de tener 11 procesos a tener 15 en total, ya que son 7 procesos finalizados, 3 procesos en cola de listos, 1 en ejecución y los 4 procesos nuevos.

Simulador de procesos

Nuevos procesos: **4**

Procesos Terminados

	Id	Op	Res
1	1	529 + 5...	1033
2	2	854 * 9...	ERROR
3	3	300 % ...	91
4	4	285 / 4...	0.6834532374100719
5	5	43 - 873	-830
6	9	470 + 7...	1176
7	6	517 * 614	317438

Listos

	Id	TiEs	TT
1	8	6	1
2	10	16	0
3	11	14	0

Proceso en Ejecución

	1
Id	7
Operacion	239 + 583
Tiempo	16
Tiempo Total	5
Tiempo Restante	11

Proceso bloqueados

ID	TB
----	----

Contador Global: **111**

Ver tiempos

Una vez terminados se nos muestra la tabla de tiempos totales.

Simulador de procesos

Nuevos procesos: **0**

Procesos Terminados

	Id	Op	Res
1	1	529 + 5...	1033
2	2	854 * 9...	ERROR
3	3	300 % ...	91
4	4	285 / 4...	0.6834532374100719
5	5	43 - 873	-830
6	9	470 + 7...	1176
7	6	517 * 614	317438
8	7	239 + 5...	ERROR
9	12	613 + 6...	1239
10	8	721 + 4...	1143
11	13	997 % ...	480
12	10	666 / 9...	0.7318681318681318
13	14	489 + 3...	815
14	11	353 - 113	240

Listos

Id	TiEs	TT
----	------	----

Proceso en Ejecución

	1
Id	15
Operacion	533 + 787
Tiempo	7
Tiempo Total	5
Tiempo Restante	2

Proceso bloqueados

ID	TB
----	----

Contador Global: **196**

Ver tiempos

Donde ahora sí podemos ver que todos los procesos tienen un estado finalizado y sus respectivos tiempos y resultados.

Tiempos										
Volver a simulador										
Id	Estado	TME	T.Llegada	T.Finalizaci	T.Retorno	T.Respuesta	T.Espera	T.Servicio	Operacion	Resultado
1	Finalizado	15	0	55	55	0	40	15	529 + 504	1033
2	Finalizado	14	0	58	58	8	53	5	854 * 946	ERROR
3	Finalizado	9	0	64	64	9	55	9	300 % 209	91
4	Finalizado	5	0	75	75	10	70	5	285 / 417	0.683453237...
5	Finalizado	13	55	88	33	9	20	13	43 - 873	-830
6	Finalizado	10	58	107	49	17	39	10	517 * 614	317438
7	Finalizado	16	64	119	55	14	42	13	239 + 583	ERROR
8	Finalizado	6	75	132	57	4	51	6	721 + 422	1143
9	Finalizado	12	88	100	12	0	0	12	470 + 706	1176
10	Finalizado	16	100	161	61	23	45	16	666 / 910	0.7318681318...
11	Finalizado	14	107	193	86	17	72	14	353 - 113	240
12	Finalizado	6	119	131	12	6	6	6	613 + 626	1239
13	Finalizado	14	131	146	15	1	1	14	997 % 517	480
14	Finalizado	5	132	166	34	29	29	5	489 + 326	815
15	Finalizado	7	146	198	52	31	45	7	533 + 787	1320

Conclusiones.

En este caso el programa no fue tan complicado de realizar ya que solo teníamos que agregar algunas funciones, y anteriormente ya habíamos hecho algo parecido que es el caso de las teclas, y cómo manejamos los procesos como objetos, no fue complicado el agregar un estado dentro de estos ya que solo se modificaban en casos muy específicos. Lo que sí se nos complicó fue el tener las validaciones necesarias para cada proceso, ya sea el caso de los procesos en la cola de listos, que tienen que mostrar algunos tiempos con el tiempo que tienen en ese momento. Ya que tuvimos que agregar varias comparaciones que nos ayudaran a saber si debía de tener algún tiempo o debía de ser un dato nulo.