UNIVERSIDAD DE GUADALAJARA

Centro Universitario de Ciencias Exactas e Ingenierías

Ingeniería en Computación

6to semestre

Tema: Paginación Simple

Materia:
Seminario de Sistemas Operativos

Sección:
D02

NRC:
103845

PRESENTA:
Saúl Alejandro Castañeda Pérez
Daniel Martínez Martínez

Docente:
Violeta del Rocio Becerra Velazquez

Fecha de entrega:
Domingo, 14 de Mayo de 2023

Representación de paginación simple

Introducción

Este programa contempla la implementación de mejoras al programa 10, el planificador Round Robin. Trabajaremos implementando la administración de memoria a través de la técnica de paginación simple, que a diferencia del programa anterior, en este se maneja una memoria de 200 unidades, con marcos de 5 unidades cada uno (40 marcos), en vez de una capacidad de memoria fija para 4 procesos.

La paginación simple es una técnica para administrar la memoria que consiste en dividir la memoria en espacios de igual tamaño, a cada espacio se le denomina como frame. Esta característica establece que para que un proceso pueda acceder a memoria, este debe dividirse en partes, a cada parte se le denomina página. El tamaño de una página debe ser igual al tamaño de un marco, porque la idea es que la página embone en el marco. Para que el proceso entre al sistema, es necesario que se almacene la totalidad de sus páginas en los distintos marcos que hay en la memoria, de lo contrario, se mantendrá en espera hasta que se libere el espacio suficiente para que el proceso entre por completo.

Se deberá mostrar la memoria, con la información del número de marco, la cantidad de espacios que se ocupan dentro del frame, que proceso ocupa dicho frame y el estado en el que este se encuentra. Además, en caso de que se muestre toda la memoria en la pantalla, al teclear la tecla “A”, funcionará como una pausa. Opté por mostrar toda la memoria en la misma interfaz debido a que considero que se ve más limpio y entendible. Finalmente, se deberán conservar con todos los requisitos del programa 5, el algoritmo de planificación Round Robin.

Desarrollo

La naturaleza de este programa plantea que el número de procesos que se admitirán en el sistema bajo el estado de listos será definido por la relación entre el tamaño de los procesos y el número de marcos disponibles que hay en memoria, debido a que los procesos son capaces de entrar solo si la memoria cuenta con los marcos suficientes para alojar a la totalidad de las páginas en las que los procesos se dividieron.

Ya estando en el sistema, los procesos son divididos en páginas y alojados en los marcos según su tamaño lo demande. Ahora como contamos con este método donde debemos visualizar los procesos divididos en fragmentos, se implementó una pequeña ventana con formato de tabla que muestra en tiempo real el comportamiento de los procesos.

En la tabla, se muestran las páginas de lo procesos dentro de los marcos, siguiendo un uso de distintas tonalidades de amarillo/naranja para distinguirlos entre si, ademas de que esto sirve para apreciar la caracteristica de que las paginas de los procesos no son almacenadas necesariamente de forma secuencial, si no que se distribuyen a lo largo de la memoria segun sea la disponibilidad de los marcos de memoria.

También se establece una coloración según el estado de procesamiento en el que se encuentre la tarea, siendo estos los siguientes

- Color verde: las celdas rellenadas con este color pertenecen al proceso que está haciendo uso del procesador, es decir, que está en ejecución. El proceso en ejecución permanecerá en ese color una cantidad determinada de tiempo de manera cíclica, dado que el planificador empleado (RR) así lo plantea.
- Tonalidad amarilla: Las celdas o marcos que cuenten con este color pertenecen a los distintos procesos que se encuentran en estado de listo.
- Color rojo: Las celdas en este color pertenecen a los procesos que han sido interrumpidos y enviados a la tabla de bloqueados. Después de que pasa el tiempo de bloqueo, vuelven a la tonalidad naranja de los listos.
- Color gris oscuro: Se reservan dos marcos completos de la memoria para alojar de manera simbólica a lo que sería el sistema operativo, dado que esa es una región de la memoria que no debe ser alterada por seguridad.
- Marcos libres: como su nombre lo indica, son los marcos que están disponibles para su uso por parte de otro proceso. Se dice que un marco está libre si todas sus celdas están desocupadas, dado que no es posible alojar partes de varios procesos en una misma página, a pesar de que se esté desperdiciando espacio de memoria.

Esta vista de la paginación aparece en todo momento en la esquina inferior derecha, sin embargo, es posible presionar la tecla ‘A’ para obtener una vista más detallada de esta característica en un momento determinado dentro de la ejecución del programa.

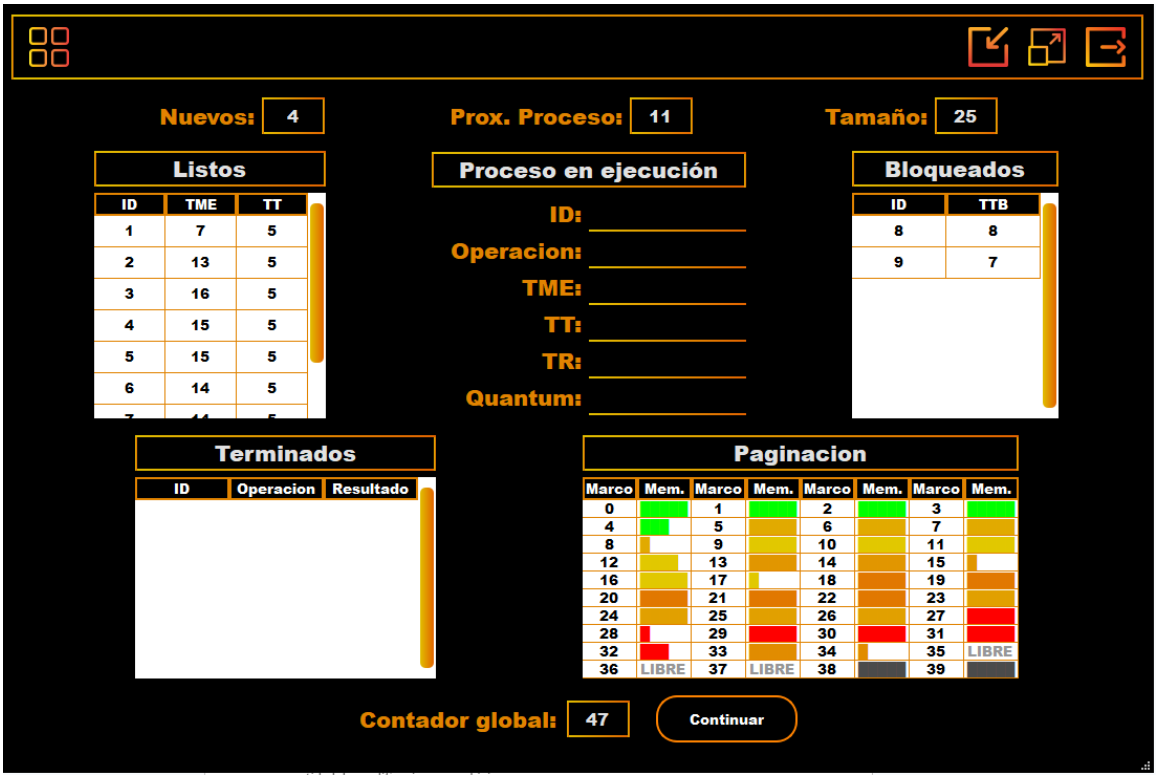
Dicha vista muestra datos adicionales como el ID del proceso que ocupa cada marco y el estado de procesamiento en el que se encuentra.

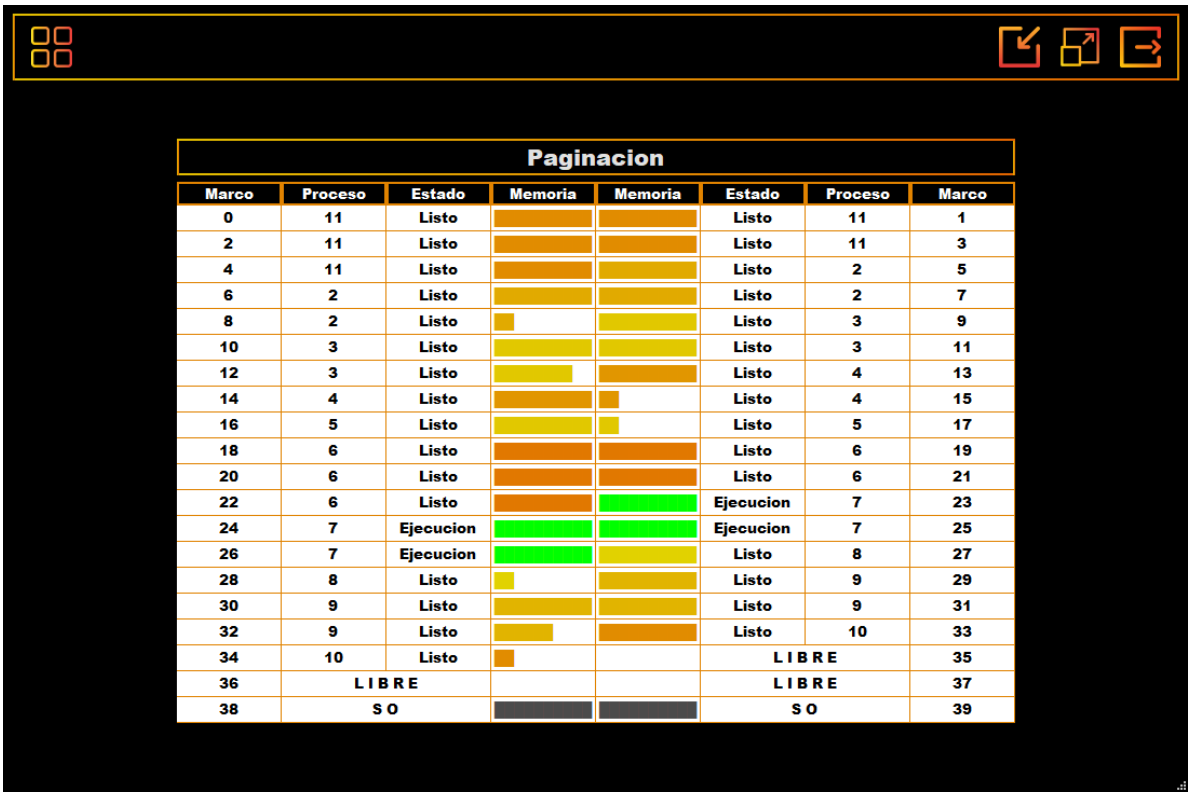
Además, en la vista principal se implementó 2 indicadores los cuales nos muestran el ID y tamaño del próximo proceso a entrar en el sistema, esto con la finalidad de ver que el programa tiene un desempeño satisfactorio.

CÓDIGO RELEVANTE

```
self.isAlgoritmoCorriendo = True
    while self.tableWidget_5.rowCount() != 0 or self.tableWidget_2.rowCount() != 0:
        if self.tableWidget_5.rowCount() == 0:
            DesbloquearProcesosBloqueados(self)
            IncrementarTE(self)
            IncrementarContadorGlobal(self)
            sleep(self,500)
            continue
        Proceso = Proceso % self.tableWidget_5.rowCount()
        ID_Proceso = self.tableWidget_5.item(Proceso, 0)
        ID_Proceso = int(ID_Proceso.text()) - 1
        if ProcesoEnEjecucion(self,self.Lotes[ID_Proceso]):
            if self.BanderaInterrupcion:
                self.tableWidget_5.removeRow(Proceso)
                self.BanderaInterrupcion = False
            else:
                item = QtWidgets.QTableWidgetItem(str(self.Lotes[ID_Proceso][self.getPos['TS']]))
                item.setTextAlignment(QtCore.Qt.AlignCenter)
                self.tableWidget_5.setItem(Proceso, 2, item)
                Proceso += 1
        else:
            self.tableWidget_5.removeRow(Proceso)
            IngresarNuevoProceso(self)
    self.isAlgoritmoCorriendo = False
```

Resultados


Nuevos: 4
Prox. Proceso: 11
Tamaño: 25
Listos
ID TME TT
1 7 5
2 13 5
3 16 5
4 15 5
5 15 5
6 14 5
Proceso en ejecución
ID:
Operacion:
TME:
TT:
TR:
Quantum:
Bloqueados
ID TTB
8 8
9 7
Terminados
ID Operacion Resultado
Paginacion
Marco Mem. Marco Mem. Marco Mem. Marco Mem.
0 1 2 3
4 5 6 7
8 9 10 11
12 13 14 15
16 17 18 19
20 21 22 23
24 25 26 27
28 29 30 31
32 33 34 35 LIBRE
36 LIBRE 37 LIBRE 38 39
Contador global: 47
Continuar


Paginacion
Marco Proceso Estado Memoria Memoria Estado Proceso Marco
0 11 Listo Listo 11 1
2 11 Listo Listo 11 3
4 11 Listo Listo 2 5
6 2 Listo Listo 2 7
8 2 Listo Listo 3 9
10 3 Listo Listo 3 11
12 3 Listo Listo 4 13
14 4 Listo Listo 4 15
16 5 Listo Listo 5 17
18 6 Listo Listo 6 19
20 6 Listo Listo 6 21
22 6 Listo Ejecucion 7 23
24 7 Ejecucion Ejecucion 7 25
26 7 Ejecucion Listo 8 27
28 8 Listo Listo 9 29
30 9 Listo Listo 9 31
32 9 Listo Listo 10 33
34 10 Listo LIBRE 35
36 LIBRE LIBRE 37
38 SO SO 39

Conclusiones

Este programa presentó una dificultad significativa, al punto de que nos vimos obligados a hacer una reestructuración general. Decidimos separar la lógica del programa en otro archivo .py para mejorar la legibilidad, ya que se había vuelto ilegible debido a la gran cantidad de modificaciones que hicimos.

Afortunadamente fuimos capaces de reciclar la mayor parte del código encargado de la lógica planteada para satisfacer los requerimientos de los programas previos, como lo es el manejo de contadores, la captación de teclas, el uso de quantum, etc.

Uno de los principales problemas que surgió fue la manera en la que llenamos los espacios en cada uno de los marcos, dado que habíamos planteado y tratado de implementar varias formas de cumplir dicha tarea, sin tener éxito. Finalmente se nos ocurrió usar caracteres especiales del ASCII, los cuales íbamos coloreando mientras los posicionamos en cada marco.