



Universidad de Guadalajara.

Centro Universitario de Ciencias Exactas e Ingenierías.

DIVISIÓN DE TECNOLOGÍAS PARA LA INTEGRACIÓN CIBER HUMANA.

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES.

TEMA: Practica3

NOMBRE DE LOS ESTUDIANTES:

Padilla Perez Jorge Daray

Juan Jesús Sámano Juárez

Ernesto Macias Flores

NOMBRE DE LA MATERIA: Seminario Sistemas Operativos

SECCION: D04

CICLO ESCOLAR: 2024-A

NOMBRE DEL PROFESOR: Julio Esteban

Contenido

Introducción.....	3
Capturas de pantalla.....	4
Código fuente.	5
Main():.....	5
Conclusiones:	7

Introducción.

En este documento que se muestra cómo se realizan acciones mediante teclas, dichas funciones como I para interrumpir procesos, E para marcar error, P para pausar el proceso y C para continuar el proceso pausado, esta actividad funciona junto con la practica2 sobre lotes así mismo se lleva una continuidad que también se refleja plasmada en este documento.

También se presentan capturas de las ejecuciones del programa funcionando de manera correcta de los momentos que creemos cruciales, el código fuente que se utilizó para el desarrollo del programa, y por ultimo las conclusiones personales de cada integrante del equipo.

Capturas de pantalla.

Código fuente.

Main():

```
1  import sys
2  import random
3  from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QHBoxLayout, QPushButton, QTableWidgetItem, QLabel, Q
4  from PyQt5.QtCore import QThread, pyqtSignal, QTimer
5  from statistics import mean
6
7
8  class ProcesoThread(QThread):
9      proceso_agregado = pyqtSignal(list)
10
11      def __init__(self, proceso):
12          super().__init__()
13          self.proceso = proceso
14
15      def run(self):
16          # Simulación de operaciones intensivas
17          self.proceso_agregado.emit(self.proceso)
18
19  class VentanaPrincipal(QWidget):
20      def __init__(self):
21          super().__init__()
22
23          self.setWindowTitle("Planificador de Procesos")
24          self.setGeometry(100, 100, 1000, 600)
25
```

Class proceso que sirve para crear los procesos automáticos y la venta también sale la clase.

```
32      def iniciarInterfaz(self):
33          # Botones
34          hbox_botones = QHBoxLayout()
35          self.nombres_botones = ["Agregar", "Terminar", "Pausar", "Continuar"]
36          for nombre in self.nombres_botones:
37              boton = QPushButton(nombre)
38              hbox_botones.addWidget(boton)
39              boton.clicked.connect(self.on_boton_clickeado)
40
41          # Botón FCFS
42          self.btn_fcfs = QPushButton("FCFS")
43          self.btn_fcfs.clicked.connect(self.iniciar_FCFS)
44
45          # Etiqueta para el ciclo
46          self.etiqueta_ciclo = QLabel("Ciclo: ---")
47          self.hbox_fcfs_ciclo = QHBoxLayout()
48          self.hbox_fcfs_ciclo.addWidget(self.btn_fcfs)
49          self.hbox_fcfs_ciclo.addWidget(self.etiqueta_ciclo)
50
51          # Tabla de procesos
52          self.tabla_procesos = QTableWidgetItem()
53          self.tabla_procesos.setColumnCount(len(self.encabezados_procesos))
54          self.tabla_procesos.setHorizontalHeaderLabels(self.encabezados_procesos)
55
```

Inicio de interfaz grafica y su enlazamiento.

```

95     def on_boton_clickeado(self):
96         sender = self.sender()
97         self.pausa = 0
98         self.terminar = 0
99         if s (variable) tiempo_llegada: str
100             self.proceso_actual) # Convertir el índice actual a una letra mayúscula
101             tiempo_llegada = str(random.randint(self.ciclo_actual, self.ciclo_actual + 5)) # Generar un tiempo de llegada aleatorio e
102             tiempo_ejecucion = str(random.randint(10, 15)) # Generar un tiempo de ejecución aleatorio entre 10 y 15
103             proceso = [proceso_nombre, tiempo_llegada, "0", "0", tiempo_ejecucion, "0", "0", "0"] # Ejemplo de proceso
104             self.agregar_proceso(proceso)
105             self.proceso_actual = (self.proceso_actual + 1) % 26 # Incrementar el índice y volver a "A" si alcanza "Z"
106         if sender.text() == "Terminar":
107             self.terminar = 1
108         if sender.text() == "Pausar":
109             self.pausa = 1
110         if sender.text() == "Continuar":
111             self.pausa = 0
112

```

Trigger para los botones al momento de estar en ejecución los procesos.

```

113     def agregar_proceso(self, proceso):
114         self.procesos.append(proceso)
115         self.agregar_proceso_tabla(proceso) # Agregar el proceso a la tabla de procesos inmediatamente
116
117     def actualizar_tabla(self, proceso):
118         row_count = self.tabla_procesos.rowCount()
119         self.tabla_procesos.setRowCount(row_count + 1)
120         for col, valor in enumerate(proceso):
121             item = QTableWidgetItem(valor)
122             self.tabla_procesos.setItem(row_count, col, item)
123         # Agregar barra de carga en la columna "Porcentaje"
124         progreso = QProgressBar()
125         progreso.setValue(0)
126         self.tabla_procesos.setCellWidget(row_count, 7, progreso)
127         self.tabla_procesos.sortItems(1) # Ordenar por la columna 1 (tiempo de llegada)
128         self.calcular_promedio()

```

Agregar procesos y actualizar tabla aparecen para que se siga ciclando.

```

147     def logica_FCFS(self):
148         # Ordenar la tabla "tabla_procesos" por la columna de tiempo de llegada
149         # Lógica específica del algoritmo FCFS
150         self.calcular_promedio()
151         for i in range(self.tabla_procesos.rowCount()):
152             proceso = [self.tabla_procesos.item(i, j).text() for j in range(self.tabla_procesos.columnCount())]
153             if self.pausa == 1:
154                 proceso[5] = str(int(proceso[5]) + 1)
155                 self.tabla_procesos.setItem(i, 5, QTableWidgetItem(str(int(proceso[5]))))
156                 proceso[6] = str(self.ciclo_actual)
157                 self.tabla_procesos.setItem(i, 6, QTableWidgetItem(str(int(proceso[6]))))
158                 break
159             elif self.terminar == 1:
160                 self.eliminar_fila_seleccionada()
161                 self.terminar = 0
162                 break
163             elif proceso[7] != proceso[4] and int(proceso[1]) <= self.ciclo_actual: # Si el proceso no ha completado su tiempo
164                 if proceso[7] == "0":
165                     self.tabla_procesos.setItem(i, 2, QTableWidgetItem(str(self.ciclo_actual)))
166                     self.actualizar_progreso(proceso, i)
167                     if proceso[7] == proceso[4]:
168                         self.tabla_procesos.setItem(i, 3, QTableWidgetItem(str(self.ciclo_actual + 1)))
169                     break
170         # Conectar la señal itemSelectionChanged a la función eliminar_fila_seleccionada

```

La lógica del FCFS se representa aquí.

Faltaría realizar las interrupciones, y también faltaría realizar la terminación de interrupciones.

Conclusiones:

Padilla Perez Jorge Daray:

Para concluir con esta actividad interesante ya que es raro tener que simular que una computadora actual tarde tanto en hacer ciertos procesos y más cuando es algo tan sencillo, el tema de tener varias ventanas fue todo un reto que el compañero Ernesto soluciono, la comunicación en equipo fue importante para la realización de esta actividad, aprendí juntos a mis compañeros y espero que podamos seguir así de comunicados.

Juan Jesús Sámano Juárez:

Al empezar a realizar esta actividad me costó trabajo entender los requerimientos, con apoyo de mis compañeros pude entender de mejor manera, en un principio realicé un código que cumplía con la mayoría de lo requerido, después con el equipo se decidió reunir 3 trabajos para hacer uno solo, aprendí sobre el manejo de hilos y a trabajar mediante GitHub.

Ernesto Macias Flores:

El desarrollo de un simulador de lotes en Python es una tarea valiosa para comprender los conceptos de planificación de procesos y la ejecución de tareas en un entorno de procesamiento por lotes, como una buena introducción a lo que se viene después, y aunque es una simulación es útil para comparar y despejarnos un poco de lo que es lo normal en la programación.