TIME RECORDING LOG

Student: José Eduardo Hernández Rodríguez Date: 5 de septiembre, 2022

Instructor: Juan Manuel Gonzales Calleros Program#: PSP0_B1_201938227.py

Date	Start	Stop	Interruption Time	Delta Time	Phase	Comments
4/9	9:35	11:18	20	17	Analysis	Ir de compras
			27			Cocinar
			2			Tomar agua
			2			Responer mensajes
			35			Comer
	11:18	23:50	55	287	Coding	Falla corriente eléctrica
	13:38		84			Falla corriente eléctrica
	17:00		10			Tomar agua
			205			Salida
	20:22		193			Falla corriente eléctrica
						Dormir
5/9	8:00	16:48	20		Coding	Desayunar
	9:00		60			Clase
	11:00		60			Clase
	13:00		60			Clase
	15:00		60			Clase
			156			Otras actividades entre clases
			30			Regresar a casa
	16:29	23:50	345	33	Pruebas	Falla corriente eléctrica
			15			Ordenar comida
			15			Bañarse
			13			Lavar trastes
			20			Comer
	00:05	01:17	15	42	Post Mortem	Realizar otras tareas
			15			Preparar café y tomar
Total			1492	379		

Post Mortem

Análisis

Programa: PSP0_B1_20193822.py

Se requiere un programa que cuente las líneas lógicas de código de un programa, omitiendo las líneas en blanco y los comentarios.

Restricciones: Usar como base el estándar de medición (R1)

Una vez codificado, se realizan las pruebas necesarias para confirmar que el sistema es correcto. Las anteriores serán descritas en el apartado de pruebas

PSEUDOCÓDIGO

 # Función read

```
WHEN archivo_name is OPEN
        content = read()
RETURN content
```

 # Funcion para contar el número de líneas de código en el archivo

```
numComments = content.count("#")
RETURN numComments
```

 # Función para contar el número de líneas en blanco

```
líneas_blanco = 0
FOR línea in content.splitlines()
        IF linea.strip() == ""
                lineas_blanco++
RETURN lineas_blanco
```

 # Función para contar el número de líneas de Código

```
líneas_codigo = content.count("\n")+1
RETURN lineas_codigo
```

 # Mostrar líneas lógicas

lineas_codigo – lineas_blanco – num_comments

Planeación:

Lenguaje por utilizar: Python versión 3.9.6

Elementos necesarios:

1. Clase File
2. Main function

Diagrama de clases

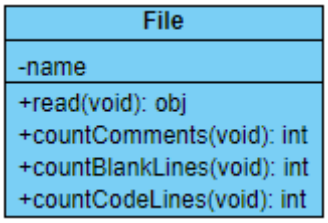
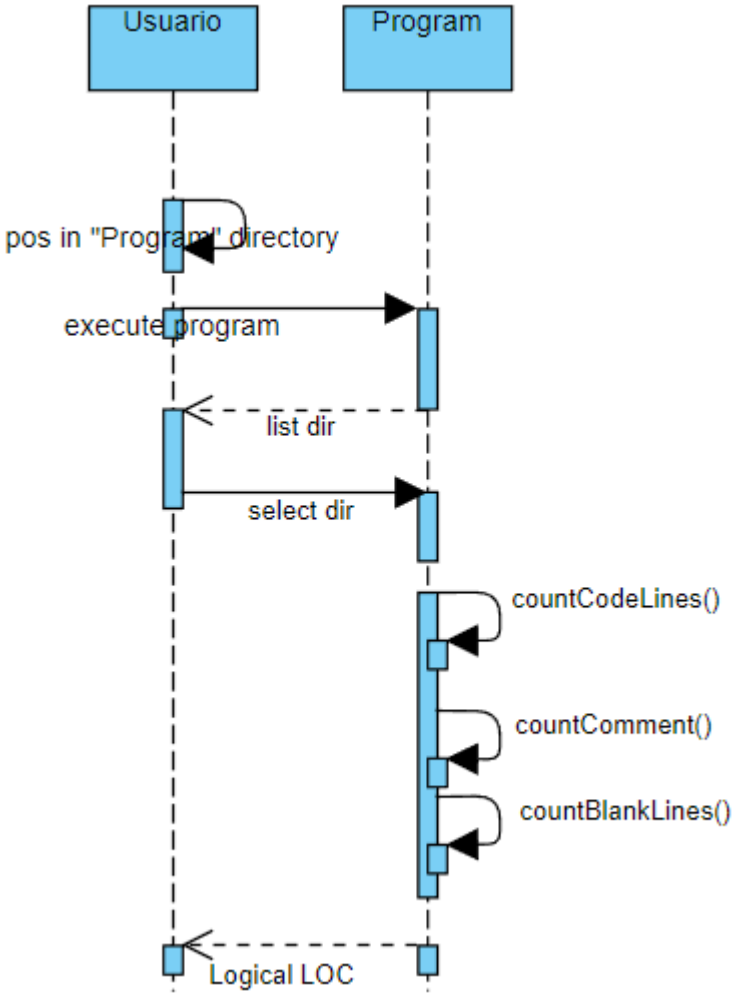


Diagrama de secuencia



Pruebas sugeridas

Contar manualmente las líneas de código de los programas 1A, 1B, 2A y B1, posteriormente ejecutar el programa y comparar los resultados obtenidos.

	Programa	LOC
A1	48	48
A2	51	51
A3	109	109
B1	30	30

Capturas de pantalla de la ejecución de las pruebas:

Posicionamiento en el directorio “Program”

```
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0> cd .\Program\
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0\Program>
```

Programa A1:

```
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0\Program> python .\PSP0_B
1_201938227.py
[0]: PSP0_201938227.py
[1]: PSP0_A2_201938227.py
[2]: PSP0_A3_201938227.py
[3]: PSP0_B1_201938227.py


Select a file: 0
Using: PSP0_201938227.py
Logical lines of code: 48
```

Programa A2:

```
Select a file: 1
Using: PSP0_A2_201938227.py
Logical lines of code: 51
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0\Program>
```

Programa A3:

```
Select a file: 2
Using: PSP0_A3_201938227.py
Logical lines of code: 109
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0\Program>
```

Programa B1

```
Select a file: 3
Using: PSP0_B1_201938227.py
Logical lines of code: 30
PS F:\Escuela\7mo Semestre\Control de calidad de software\Program_B1 PSP0\Program>
```