

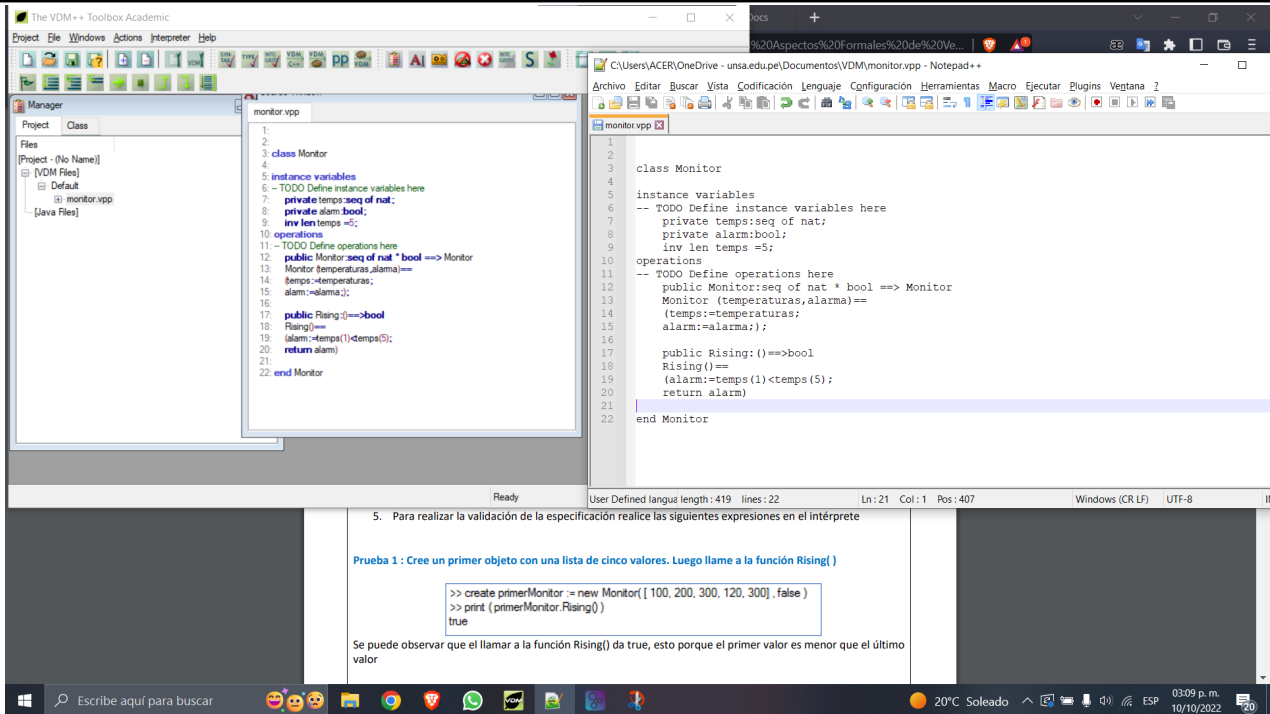
	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 1</p>

INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	ASPECTOS FORMALES DE ESPECIFICACIÓN Y VERIFICACIÓN				
TÍTULO DE LA PRÁCTICA:	<i>Creación de Clases e Invariantes</i>				
NÚMERO DE PRÁCTICA:	<i>03</i>	AÑO LECTIVO:	2022	NRO. SEMESTRE:	2022B
FECHA DE PRESENTACIÓN	<i>10-10-2022</i>	HORA DE PRESENTACIÓN	<i>04:10 p.m.</i>		
INTEGRANTE (s): GRUPO "B1 " <ul style="list-style-type: none"> Chaco Huamani, Alex Cozco Mauri, Yoset Flores Valdivia, Denilson Edgardo Tito Durand, Rudy Roberto 				NOTA:	
DOCENTE(s): <ul style="list-style-type: none"> Mg. Maribel Molina Barriga 					

SOLUCIÓN Y RESULTADOS
<p>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</p> <ol style="list-style-type: none"> En el editor de texto Notepad ++ copiamos la siguiente especificación VDM++ y grabamos como Monitor.vpp Creamos un nuevo proyecto en la Herramienta y cargue el archivo. Verificamos que no existen errores de Sintaxis y que no existen errores de Tipo



The screenshot shows the VDM++ IDE with the 'monitor.vpp' file open. The code defines a 'Monitor' class with instance variables 'temps' and 'alarm', and operations 'Rising()'. The interpreter window at the bottom shows the following commands and output:

```

5. Para realizar la validación de la especificación realice las siguientes expresiones en el intérprete

Prueba 1 : Cree un primer objeto con una lista de cinco valores. Luego llame a la función Rising()

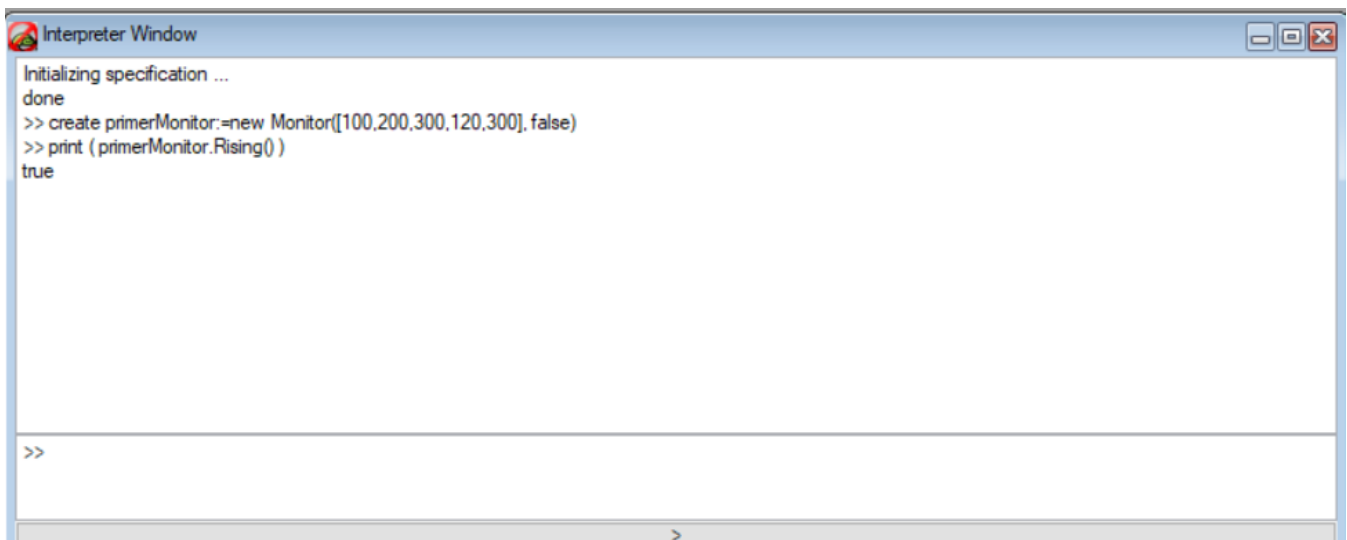
>> create primerMonitor := new Monitor([100,200,300,120,300], false)
>> print (primerMonitor.Rising())
true

Se puede observar que el llamar a la función Rising() da true, esto porque el primer valor es menor que el último valor

```

5. Para realizar la validación de la especificación realice las siguientes expresiones en el intérprete

1. Prueba 1 : Cree un primer objeto con una lista de cinco valores. Luego llame a la función Rising()
 - >> create primerMonitor:=new Monitor([100,200,300,120,300], false)
 - >> print (primerMonitor.Rising())

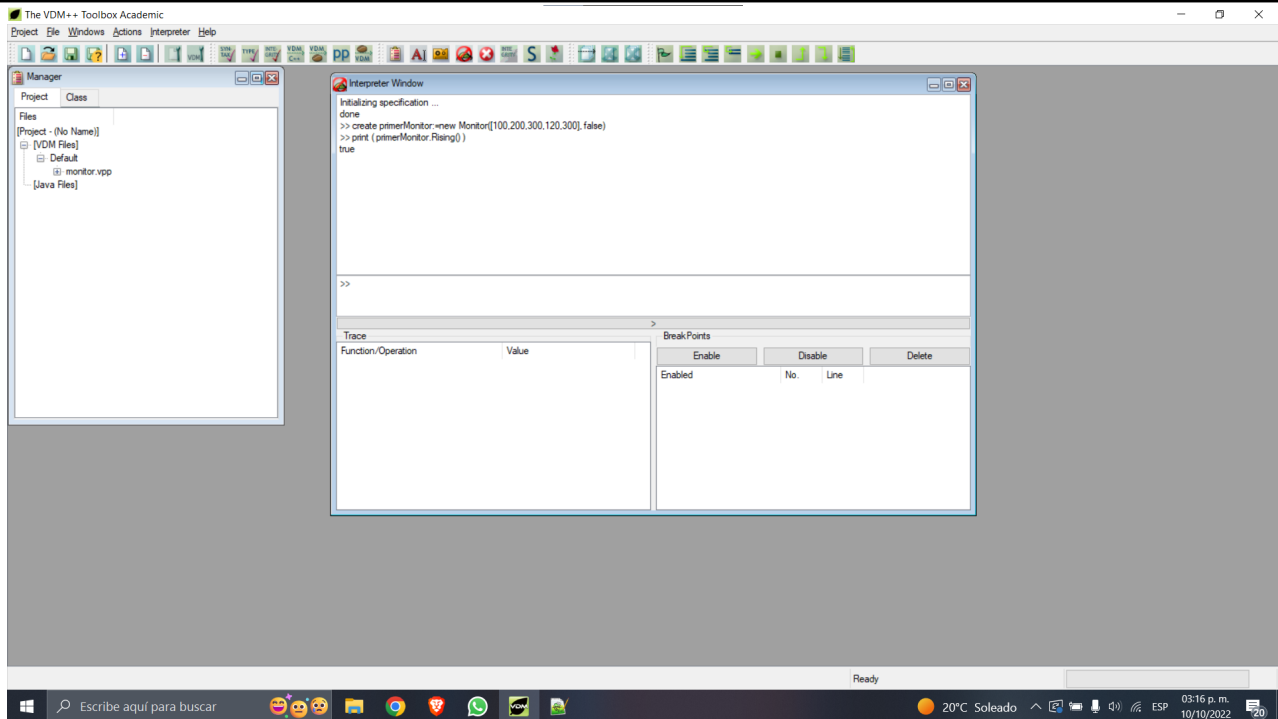


The screenshot shows the 'Interpreter Window' with the following text:

```

Initializing specification ...
done
>> create primerMonitor:=new Monitor([100,200,300,120,300], false)
>> print ( primerMonitor.Rising() )
true
>>

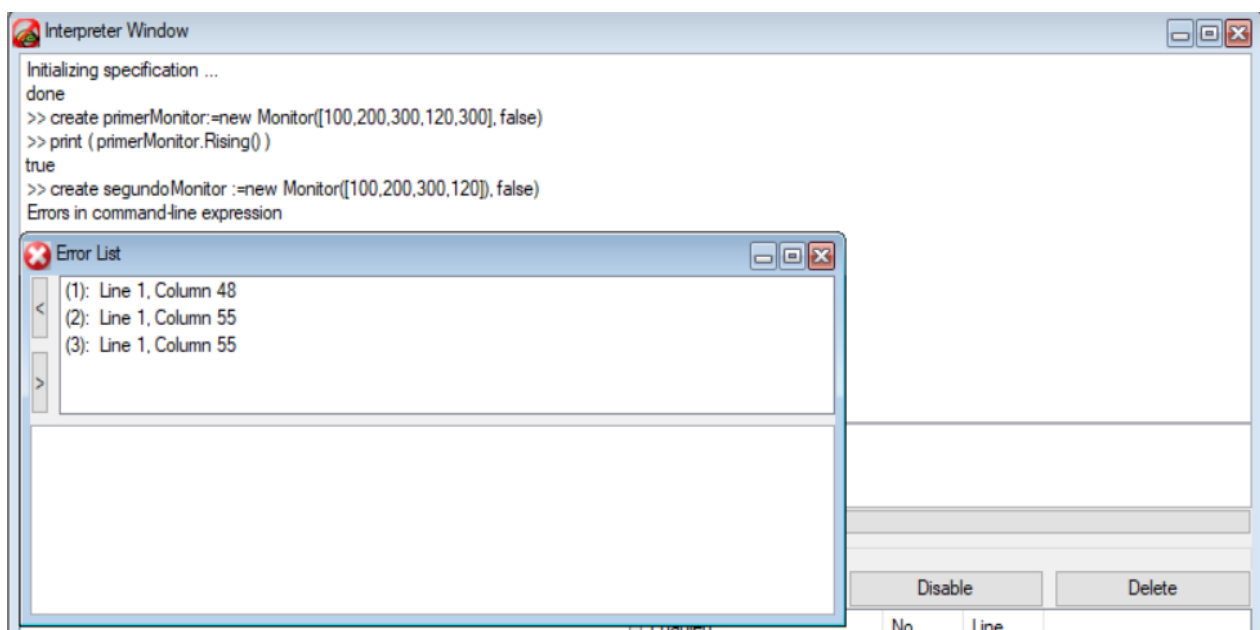
```



Se puede observar que el llamar a la función Rising() da true, esto porque el primer valor es menor que el último valor.

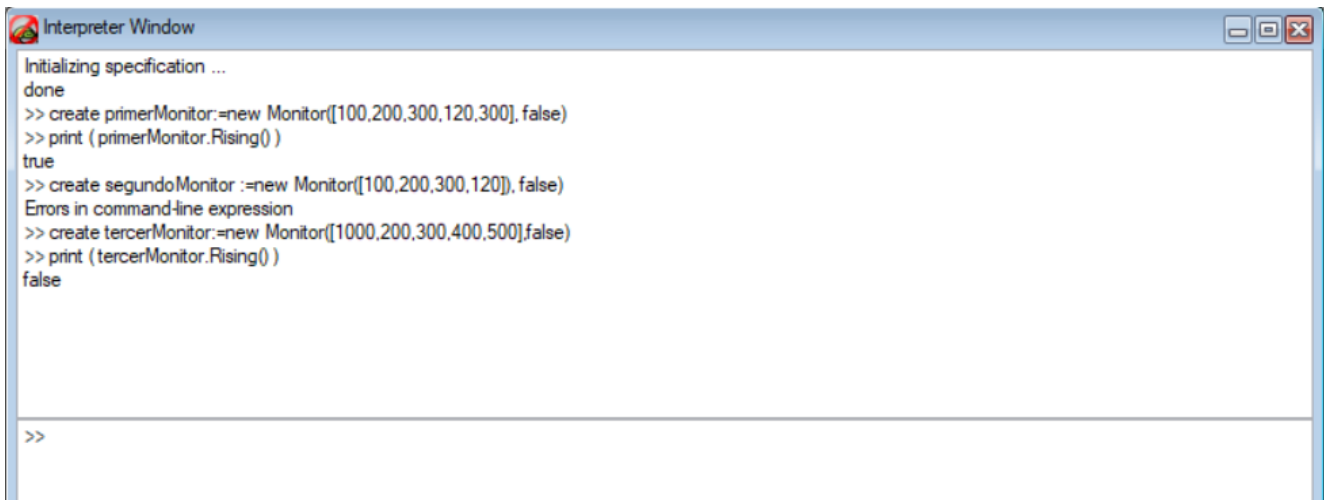
2. Prueba 2: Cree un segundo objeto con una lista con solo cuatro valores. Usted observará un mensaje que indica que la Invariante no se cumple

- >> create segundoMonitor :=new Monitor([100,200,300,120]), false)



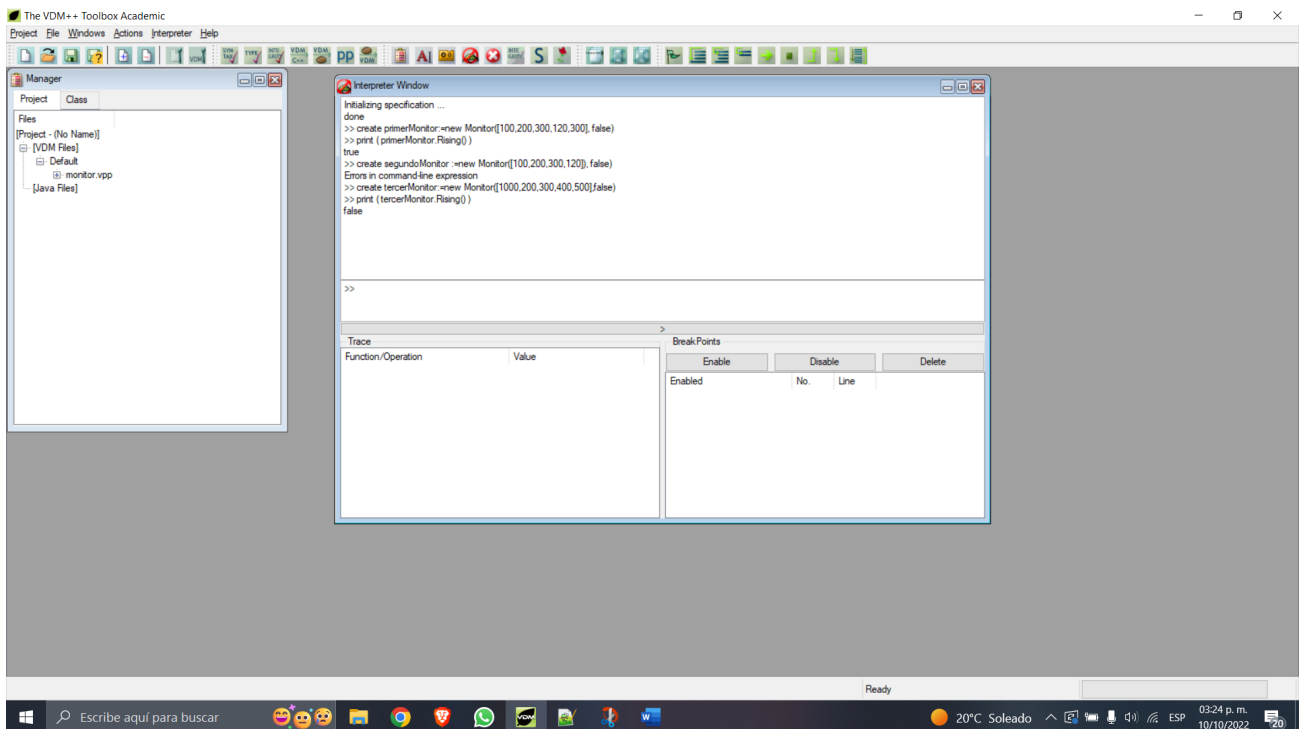
3. Prueba 3: Cree un tercer objeto con una lista de cinco valores. Luego llame a la función Rising()

- >> create tercerMonitor:=new Monitor([1000,200,300,400,500],false)
- >> print (tercerMonitor.Rising())



```

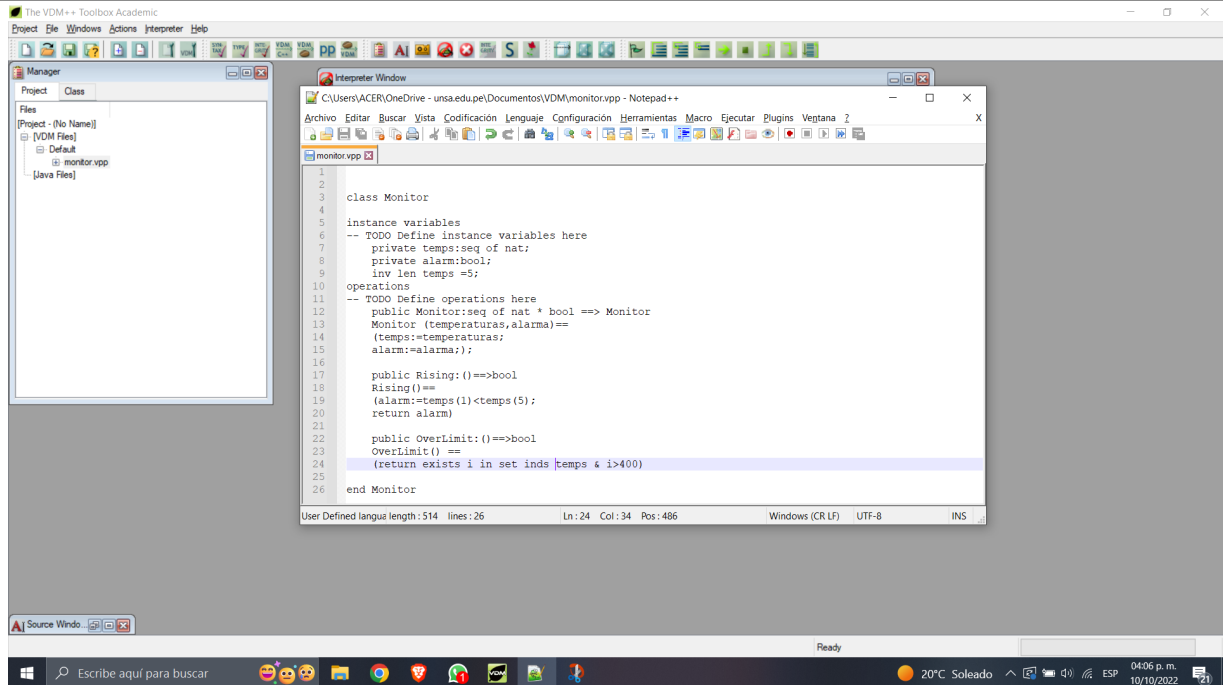
Interpreter Window
Initializing specification ...
done
>> create primerMonitor:=new Monitor([100,200,300,120,300], false)
>> print ( primerMonitor.Rising() )
true
>> create segundoMonitor :=new Monitor([100,200,300,120]), false)
Errors in command-line expression
>> create tercerMonitor:=new Monitor([1000,200,300,400,500],false)
>> print (tercerMonitor.Rising() )
false
>>
  
```



EJERCICIOS PROPUESTOS

1. OverLimit() :

límite: hay una lectura en la muestra superior a 400 C

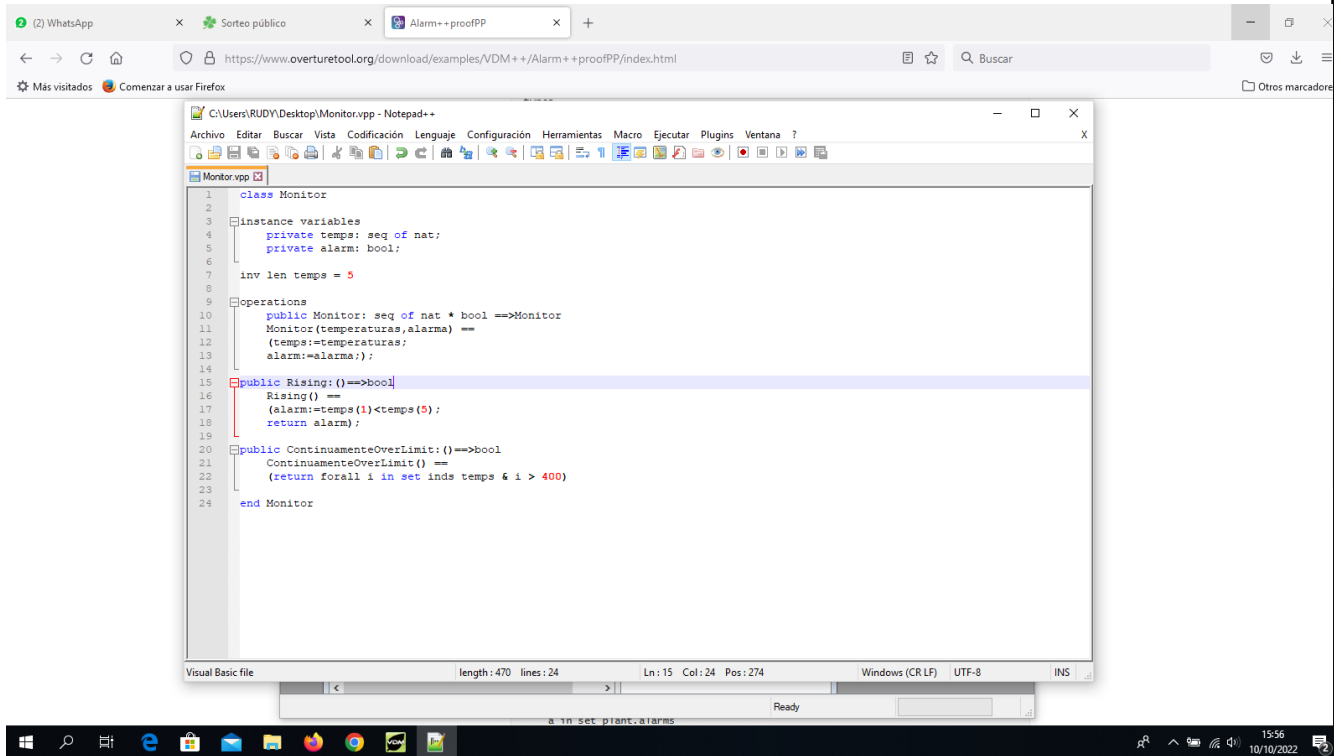


```

1
2
3 class Monitor
4
5 instance variables
6 -- TODO Define instance variables here
7 private temps: seq of nat;
8 private alarm: bool;
9 inv len temps = 5;
10
11 operations
12 -- TODO Define operations here
13 public Monitor: seq of nat * bool ==> Monitor
14 Monitor(temperaturas, alarma) ==
15 (temps:=temperaturas;
16 alarm:=alarma);
17
18 public Rising(): bool
19 Rising() ==
20 (alarm:=temps(1)<temps(5);
21 return alarm)
22
23 public OverLimit(): bool
24 OverLimit() ==
25 (return exists i in set inds temps & i>400)
26
27 end Monitor
  
```

ContinuamenteOverLimit() : Todas las lecturas en la muestra superan los 400 C.

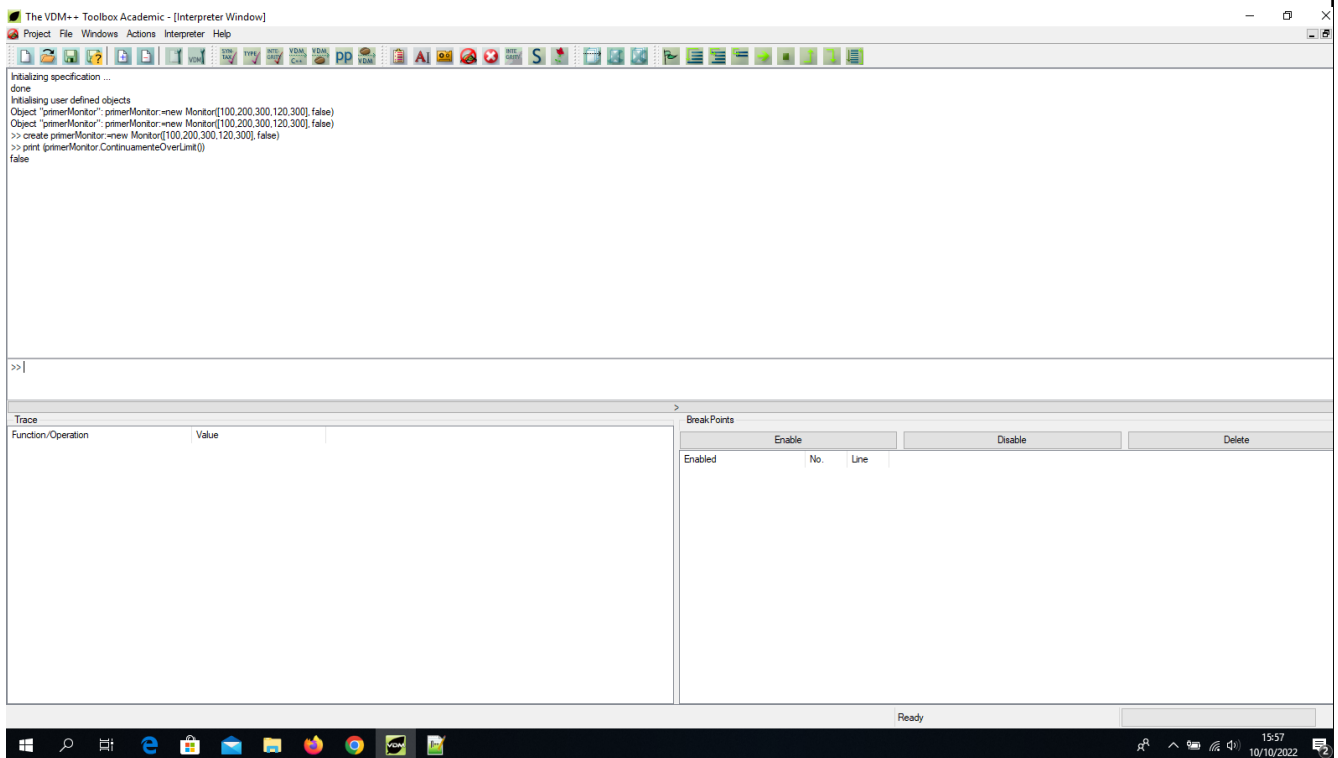
Se diseña la función **ContinuamenteOverLimit()** :



```

1
2 class Monitor
3
4 instance variables
5 private temps: seq of nat;
6 private alarm: bool;
7 inv len temps = 5
8
9 operations
10 public Monitor: seq of nat * bool ==> Monitor
11 Monitor(temperaturas, alarma) ==
12 (temps:=temperaturas;
13 alarm:=alarma);
14
15 public Rising(): bool
16 Rising() ==
17 (alarm:=temps(1)<temps(5);
18 return alarm);
19
20 public ContinuamenteOverLimit(): bool
21 ContinuamenteOverLimit() ==
22 (return forall i in set inds temps & i > 400)
23
24 end Monitor
  
```

Se hacen las pruebas



Devuelve negativo, porque ninguna temperatura pasa los 400 C.

II. SOLUCIÓN DEL CUESTIONARIO

1. Describe los tipos de básicos y compuestos del VDM++
 VDM++ cuenta con dos clasificaciones respecto a los tipos de datos: Tipos Básicos y Tipos Compuestos.
 - a. Tipo Basicos
 En esta clasificación se tiene :

Tipo Básico	Sintaxis
Booleanos	bool
Naturales	nat, natl
Reales	real

Caracter

char

b. Tipo Compuestos

En esta clasificación se tiene (entre otros):

Tipo Compuesto	Sintaxis
Conjuntos	set of
Secuencias	seq of
Mapeos	map to

2. Explique que es una invariante, mencione dos ejemplo

Son las propiedades que cada entidad (objeto) debe preservar, análogas a las "restricciones" en la base de datos. En caso de ser necesario, es posible restringir los valores en una clase a través de instancias específicas.

Para especificar una invariante se

utiliza la palabra reservada `inv` a continuación de la definición de todas las variables de instancia declaradas en la clase.



Ejemplo de Invariante:

- `inv k-Student (nombre, id, cursoconcluido, cursomatriculado) Δ cursoconcluido \wedge cursomatriculado = {}`

-

```
9      inv
10      :   latitud <= 90;
11      inv
12      :   latitud >= -90;
13      inv
14      :   longitud <= 180;
15      inv
16      :   longitud >= -180;
```

3. ¿Por qué debemos de especificar propiedades y verificar en los métodos formales?

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>

La verificación formal fuerza al análisis detallado de los requerimientos, ya que puede revelar errores potenciales que no podrían ser revelados de otra forma. Además, la especificación de propiedades reduce significativamente el número de fallos durante la ejecución del mismo.

III. CONCLUSIONES

- Se puede decir que la abstracción de las pruebas de simulación donde el refinamiento de los datos incluye una especificación para diferentes fines, como la maximización de procesamiento, a través de las máquinas de estado. Esta estructura lógica abstracta de funcionamiento es utilizada a través de la manipulación del uso de la lógica a través del código.
- El uso de métodos formales en la etapa de análisis del ciclo de vida del software aumenta el tiempo de desarrollo pero garantiza un ahorro en el tiempo de la producción facilitando la detección de errores en etapas tempranas haciendo más confiable el software que se está desarrollando.
- La herramienta utilizada, VDM++, utiliza todas las condiciones definidas en invariantes, precondiciones, poscondiciones, evitando violaciones de integridad e inconsistencia con las especificaciones formales diseñadas.

RETROALIMENTACIÓN GENERAL

REFERENCIAS Y BIBLIOGRAFÍA

- [1] https://federwin.sip.ucm.es/sic/investigacion/publicaciones/pdfs/Tesis_CarlosMoliner.pdf
- [2] https://software.imdea.org/~mcarro/Material/Formal_Methods/Formal_Methods_Intro/formal-methods_3.pdf
- [3] http://www.laccei.org/LACCEI2017-BocaRaton/student_Papers/SP484.pdf
- [4] <https://1library.co/document/zq600p8q-metodo-vdm.html>
- [5] <https://dialnet.unirioja.es/servlet/articulo?codigo=7849187>