Mail a grupo

[fsotuyo@gmail.com](mailto:fsotuyo@gmail.com)

Obligatorio

Implementar un bed lapadulla básicamente

Se van a recibir un conjunto de instrucciones y nosotros vamos a tener que validar la sintaxis según bed lapadulla, la semántica y la veracidad o correctitud con respecto a programa

Las instrucciones Read, Write, créate y destroy que se necesitan en este oblgatorio

Read Sujeto Objeto

Write Sujeto, objeto y valor

Créate sujeto objeto

Destroy

App de consola

Tomar parámetros de la consola

Parte 1

Parte 1.1

Filtrado por sintaxis, verbo bien escrito en mayúsculas

En el archivo de salida debe indicar “badinstruction” en el caso de que la sintaxis

Parte 1.2

Semantica

Se debe verificar que los objetos existen (que hayan sido creados previamente)

Parte 1.3

Aplicar BLP (Bell – Lapadulla)

Bell Lapadulla dice que

Se puede leer en un nivel hasta el mio y se pued escribir a partir de mi nivel hacia arriba

Cada vez que falla cualquiera de los 3 controles (sintaxis, semántica y correctitud) se debe generar un log de badinstruction

Bell lapadulla dice que la información fluye de un nivel inferior hacia superior

No permite que un objeto con etiqueta alta escriba en un objeto de etiqueta baja

Se compara con el sistema militar… donde el general la da una orden al soldado

La comparación es con instrucciones escritas militares

Un general puede leer todo lo que hay hacia el soldado raso

Un soldado raso puede escribir (recibir órdenes) de todos los niveles superiores, pero no de los de debajo de el

26/11/19

**Parte 1**

Archivo= InstructionList.txt

Verbo, Parametro1, Parametro2, Parametro3

Controles

1. Control de sintaxix
2. VerboValido
3. Primer parámetro Sujeto
4. Segundo parámetro Objeto
5. Tercer parámetro Valor
6. Si no es correcto, se crea un log de “badinstruction”, el control se debe hacer por comando
7. Crear función Read -
8. Crear función write
9. Crear control de semántica
   1. Sujeto existe?
   2. Objeto existe?
   3. Sujeto tiene propiedades sobre el objeto? (leer o grabar)
   4. Si sujeto NO tiene permisos sobre objeto para hacer la acción adecuada

Propone hacer 4 pasos

1. Leer el archivo
2. Verificar Sintaxis
3. Verificar Semántica
4. Procesar

En lo personal, creo que los pasos 1 y 2 deberian hacerse en conjunto y luego el procesamiento debería ir contra los atributos de bell lapadula

Parte 2 – Romper Bell Lapadula mediante Canal Encubierto

Se incorporan 3 funciones

Create

Destroy

Run

Según BLD

R= Leo hasta mi nivel

W=Escribo desde mi nivel

No hay que olvidar que BLD se creó cuando no había “re” escritura, se escribe o se destruye (como las hojas en maquinas de escribir)

La información sube de niveles bajos a niveles bajo

Sujeto H – Read en O.H

Sujeto L – Read – O.L

La “gracia” es ver quien crea el objeto que se va a usar para transmitir información de un lado al otro.

Lile SIEMPRE ejecuta lo mismo, la idea es que se de cuenta en función de lo que hizo, cual es el valor que le envía HALL, que es lo que el interpreta, si es un cero o un uno.

Como enviar un 1 o un 0 ¿ Hall a Lile

H.S. –

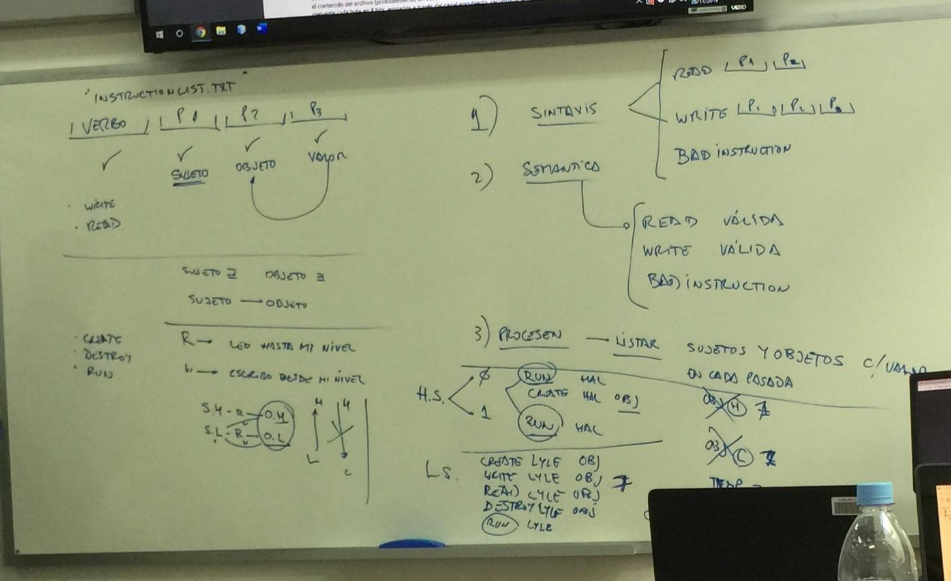
Podria ser que…

El objeto existe

---

El envio de los mensajes se deben correr en procesos independientes

Debe haber “algo” que indique que HALL no envie nada nuevo si no hubo interaccion de Lyle… se debe indicar que ya fue leído por Lyle



Run debe verificar si debe hacer o no y verificar si Lyle ya lo leyó

Y el Run de Lyle debe darse cuenta si está bien que lea o es que Hall no me mando nada todavía

* + 1. Leer “mensaje.txt”
    2. Convertirlo a bits (poner los bits uno atrás del otro)
    3. Ejecutar la transmisión entre Hall y Lyle
    4. Lyle debe ser capaz de rearmar el mensaje de Lyle

El archivo “secuencia.txt” tiene la forma en que se deben ejecutar las instrucciones. Si es H ejecuta Hall si es L ejecuta Lyle

Al leer la línea, se debe agregar un carácter de final de línea para que se rearme el texto al final del proceso.

Que se debe poner en el run de Hal y en el run de Lyle

Se entregarán 2 proyectos

1. Con la parte 1 completa de forma independiente, que lea