a) Errores:

* Igualdad Observacional:

Hay un error de redacción, porque para que dos pilas *a* y *b* sean iguales, se tienen que cumplir 2 condiciones, y son las siguientes:

- Las longitudes de *a* y *b* son iguales.

- Cada elemento en *a* es igual al correspondiente elemento en *b*.

* Falta un Género.
* Observadores básicos:

Hay un error de redacción con el segundo observador, debe ser:

es\_vacía(Pila<a>) -> Bool

* Generadores:

Como falta un Género, falta, por ejemplo:

G1() -> elGénero

* Otras operaciones:

Hay un error al incluir el observador básico es\_vacía(Pila<a>) -> Bool, debería ser:  
apilar(Pila<a>, a) -> None {Pos: la pila no está vacía)

* Exporta:

Hay un error al incluir un Usa (Natural) ya que acá solamente se indican todos los géneros, observadores básicos, generadores y todas las operaciones que puedan ser

utilizadas desde afuera.

b) Versión corregida:

* TAD Pila<a>
* Igualdad Observacional: Si *a* y *b* son dos pilas, *a* es igual a *b* si se cumple que:
* Las longitudes de *a* y *b* son iguales.
* Cada elemento en *a* es igual al correspondiente elemento en *b*.
* Usa: Natural, Bool, Secuencia<a>
* Parámetro Formal: a
* Género: Pila<a>
* Observadores básicos:
* tamaño(Pila<a>) -> Natural
* es\_vacía(Pila<a>) -> Bool
* tope(Pila<a>) -> a

{Pre: La pila tiene al menos un elemento)

* Generadores:
* G1() -> elGénero
* G2(LoQueSea) -> elGénero
* vacía() -> Pila(<a>) {Post: La pila retornada está vacía}
* a\_partir\_de(Secuencia<a>) -> Pila<a> {Post: La pila contiene apilados los elementos de la secuencia recibida}
* Otras operaciones:
* apilar(Pila<a>, a) -> None {Pos: La pila está vacía}
* desapilar(Pila<a>) -> Pila<a> {Pre: La pila tiene al menos un elemento} – {Pos: La pila perdió el tope que tenía antes de desapilar}
* Axiomas:
* vacia(): Crea una pila (sin elementos).
* a\_partir\_de(Secuencia<a> s): Crea una pila con los elementos de la secuencia s siendo el último elemento de la secuencia el que quede en el tope de la pila.
* apilar(Pila<a>, a elem): Apila en el tope de p el elemento elem.
* desapilar(Pila<a> p): Quita el elemento que se encuentra en el tope de p.
* tamaño(Pila<a> p): Retorna la cantidad de elementos de la pila p.
* es\_vacía(Pila<a> p): Retorna verdadero si la pila p está vacía y falso en caso contrario.
* tope(Pila<a> p): Retorna el elemento que se encuentra en el tope de la pila p.
* Exporta: Pila<a>, vacía, a\_partir\_de, es\_vacía, tope, tamaño, apilar, desapilar