Ejercicios tipo prueba

Paulo González

17 de septiembre de 2019

1. Resolución en lógica de predicados

1. Los siguientes axiomas definen en parte a la suma y al predicado $<$ en los número naturales:

   - $(\forall x) suma(x, 0, x)$
   - $(\forall x, y, z) suma(x, y, z) \supset suma(x, s(y), s(z))$
   - $(\forall x) menor(0, s(x))$
   - $(\forall x, y) menor(x, y) \supset menor(s(x), s(y))$

   $suma(x, y, z)$ significa que $z = x + y$, $menor(x, y)$ significa $x < y$, y $s(x)$ significa $x + 1$.

   Demostrar usando resolución a partir de los axiomas anteriores

   $$(\forall x)(\exists y) suma(x, s(s(0)), y) \wedge menor(0, y)$$

2. Algunos axiomas que describen una pila de elementos para un tipo cualesquiera, viene dado por los siguientes axiomas:

   - $(\forall x, y, p, s) top(x, p, s) \supset \neg top(x, p, pop(p, s))$
   - $(\forall x, y, p, s) push(y, p, push(x, p, s)) \supset top(y, p, pop(p, s))$
   - $(\forall p, s) esVacia(p, s) \supset esVacia(p, pop(p, s))$
   - $\neg esVacia(p, s_0)$

   Demostrar que $push(b, p, push(a, p, s_0)) \supset \neg top(a, p, pop(p, pop(p, s_0))) \wedge \neg esVacia(p, s_0)$

   Aclaración:

   Una Pila, es una estructura tipo LIFO (Last In First Out), esto quiere decir que independiente del tipo de datos, la información es accesada desde el último ingresado. Para el caso de la Pila de la figura 1 el último elemento insertado (e_n) se encuentra en el tope o cima.

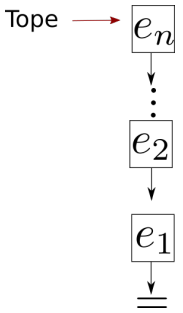

Figura 1: Pila de elementos, relación entre elementos y el acceso a éstos respecto la posción del "Tope".

La figura 1, presenta una representación de una Pila de n elementos. Comenzando desde el estado vacío, los elementos fueron insertados desde e_1 hasta e_n, y el acceso a ellos viene dado desde e_n hasta e_1, o sea, en orden inverso al ingreso.

La creación de la pila conlleva la generación de una estructura vacía, la cual no contiene elementos. La operación *push*, permite ir construyendo la pila al insertar nuevos elementos. La figura 2, muestra un ejemplo de adicionar en la pila p un nuevo elemento e. Para adicionar un elemento, se utiliza la operación *push*.

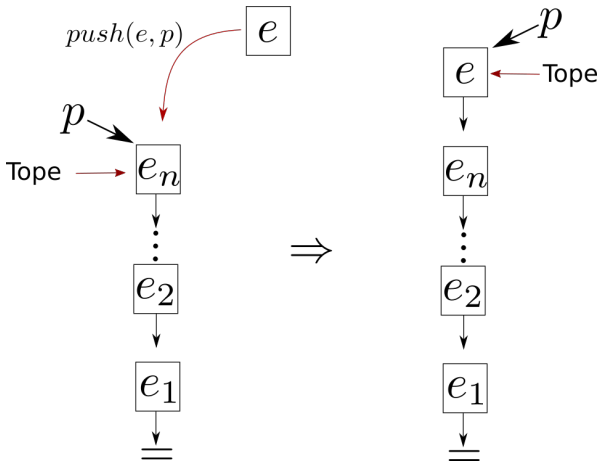

Figura 2: Adición de un elemento e en la pila p utilizando la operación *push*.

La operación *pop*, elimina el elemento del tope, y el tope apuntará al elemento anteriormente insertado. Esta operación podrá realizarse mientras la pila siga teniendo elementos. Cuando la pila esté vacía, la operación *pop* no causará ningún efecto, dado que el estado de la pila vacía no cambia.

La operación *top*, devuelve el valor del elemento. En el caso de la pila de la figura 2, la operación *top* devolverá el valor e. En el caso que la pila esté vacía, esta operación generará un error, dado que la pila no contiene elementos.

La operación $esVacia$, permite determinar si la pila contiene o no elementos.

2. Prueba formal de programa

1. El siguiente algoritmo permite restar un entero n a otro m.

```
x = m
y = n
while( x > 0)
{
   x = x - 1;
   y = y - 1;
}
```

Demostrar que el algoritmo está correcto. La precondicion es que $n \geq 0$ t la postcondición es $x = m - n$. Un invariante apropiado para este problema es $x - y = m - n \wedge y \geq 0$