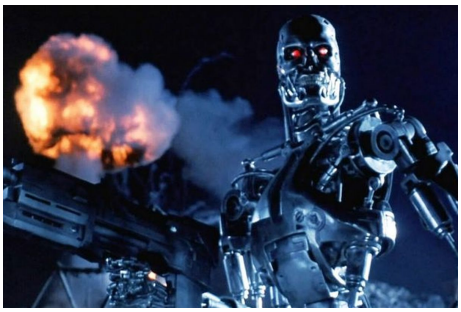
Specificator

26 de septiembre de 2018



Marty, ahora que tienes una colección de predicados, enséñales a tus padres cómo usarlos para especificar sus programas. ¡Has de convencerles de que es esencial tener claro lo que quieres que hagan tus programas, . . . o podrían rebelarse!

Descarga el fichero `Specificator.dfy`. *Crea un fichero nuevo en VSCode con el nombre de tu avatar (sin espacios) y extensión* `dfy` *y ve copiando en él cada ejercicio a medida que los vas haciendo. Has de entregar ese fichero en la tarea correspondiente del CV.* Modifica la cláusula `include` para incluir tu fichero de predicados.

1. Métodos sobre arrays: uso de cuantificadores

1. El primer ejemplo consiste en la especificación de un método `mdistintosDe` que determine si todos los elementos de un array de enteros son distintos de uno dado. Observa que admite cualquier array de entrada y en la postcondición se utiliza el predicado `distintosDe`.

2. Especifica un método `mpositivos` que determine si todos los elementos de un array de enteros son positivos utilizando el predicado `positivos` o `positivosA`, ya definidos.

3. Especifica un método `miguales` que determine si todos los elementos de un array de enteros son iguales utilizando el predicado `iguales` o `igualesA`, ya definidos.

4. Especifica un método `mparesEntreA` que determine si todos los elementos de un array de enteros contenidos entre dos posiciones dadas c (incluida) y f (excluida) son pares. Se considera que el intervalo $[c, f)$ es vacio cuando $c = f$. Recuerda, en la precondición has de colocar las condiciones que deben cumplir c y f, y también los elementos del array (usa el predicado `positivos`).

5. Especifica un método `mparesEntreC` que determine si todos los elementos de un array de enteros contenidos entre dos posiciones dadas c (incluida) y f (incluida) son pares. Se considera que el intervalo $[c, f]$ es vacio cuando $c = f+1$. Recuerda, en la precondición has de colocar las condiciones que deben cumplir c y f, y también los elementos del array (en este caso no utilices el predicado `positivos`).

6. Especifica un método `mprimerCero` que dado un array de enteros devuelva la posición en la que se encuentra el primer elemento igual a 0, y en caso de no haber ninguno devuelva la longitud del array. Utiliza el predicado `esPrimerCero`.

7. Especifica un método `mprimerNegativo` que dado un array de enteros devuelva un booleano que indica si existe algún valor negativo en el array, y en caso de haberlo la posición mas a la izquierda que contiene un valor negativo.

8. Especifica un método `mmaximo` que calcule una posición del máximo de un array de enteros. Fíjate en que se dice üna posición", ya que el máximo podría estar repetido muchas veces; a este método se le pide una de ellas, sin mostrar preferencia por ninguna en particular. Recuerda, para que el máximo esté definido al menos debe haber un elemento en el array.

9. Especifica un método `mprimerMaximo` que calcule la posición de la aparición mas a la izquierda del máximo de un array de enteros. Observa que ahora queremos la primera posición del máximo, no cualquiera.

10. Especifica un método `multimoMaximo` que calcule la posición de la aparición mas a la derecha del máximo de un array de enteros. Observa que ahora queremos la última posición del máximo.

2. Métodos numéricos

11. El primer ejemplo consiste en la especificación de un método `mdivide` que calcula el cociente y el resto de la division de naturales.

12. Especifica un método `mraiz` que calcule la raiz entera de un entero positivo.

13. Especifica un método que calcule la potencia de 2 elevado a un numero natural dado. Como la potencia de un número no es una función matemática predefinida en Dafny, tienes disponible una función de especificación potencia definida por Doc para ayudarte.

3. Métodos sobre arrays: uso de operadores

Los cuantificadores numéricos que utilizamos en los apuntes (suma, producto, contador) no están predefinidos en Dafny, pero Doc, que es muy generoso, ha definido para ti

- un operador `sumS(s)` que dada una secuencia `s` suma los elementos de dicha secuencia
- un operador `sumV(v,c,f)` que dado un array `v` y dos posiciones `c` y `f` suma los elementos del array situados en las posiciones $[c, f)$.
- un ejemplo de un operador para contar elementos de una secuencia que cumple una determinada propiedad.

Utilizalos para escribir las siguientes especificaciones:

14. El primer ejemplo es la especificación de un método que calcula la suma de los elementos de un vector, observa cómo utilizar los operadores `sumS` y `sumV`.

15. El segundo ejemplo es la especificación de un método que cuenta el numero de pares en un array de enteros positivos. Doc ha definido una función `ContarPares` que cuenta el número de pares en una secuencia.

16. Especifica un método que indica si un vector es Gaspariforme (según la definición dada en el ejercicio 8 de los apuntes del Tema 2).

17. Especifica un método que indica si hay la misma cantidad de números pares que de impares en un array de enteros positivos.

4. Métodos que modifican arrays

18. El primer ejemplo es la especificación de un método que *positiviza* un vector de enteros, lo cual consiste en reemplazar los valores negativos por cero respetando el resto de valores. Fíjate en el uso de la funcion `old` que permite hacer referencia en la postcondición a los valores que contenía el vector antes de ejecutar el método.

19. Especifica un método que en un vector de enteros reemplaza todas las apariciones de un número dado `x` por otro `y` respetando el resto de valores.

20. Especifica un metodo que dado un vector `v` de enteros devuelva otro vector `w` con su imagen especular.