EJERCICIOS TEMA 3

1. Encontrar precondiciones que garanticen el comportamiento correcto, respecto a las postcondiciones dadas, de las asignaciones siguientes. Todas las variables son de tipo natural salvo b que es booleana.

a) $\{?\}\ n := 2 * n\ \{n \leq 100\}$
b) $\{?\}\ n := n + 1\ \{n = n + 1\}$
c) $\{?\}\ n := n + 1\ \{\forall i\text{: } 1 \leq i \leq n\text{: } i \neq j\}$
d) $\{?\}\ n := i - n\ \{\forall j\text{: } 1 \leq j \leq 10\text{: } j \neq n\}$
e) $\{?\}\ n := i - n\ \{\forall i\text{: } 1 \leq i \leq 10\text{: } i \neq n\}$
f) $\{?\}\ b := b \vee (k = 0)\ \{-b\}$

2. Encontrar las precondiciones apropiadas para las siguientes asignaciones múltiples:

a) $\{?\}\ \langle i, j\rangle := \langle j, i\rangle\ \{i = j\}$
b) $\{?\}\ \langle i, j, k\rangle := \langle j, k, i\rangle\ \{i + j = k\}$

3. Derivar una instrucción alternativa que cumpla la siguiente especificación:

$\{\text{Pre: } a = A\}$
$\{\text{Post: } (x = 0 \wedge a = A + 1) \vee (x \neq 0 \wedge a = A + x)\}$

4. Supongamos que la variable *a* se ha declarado como un vector de naturales indexado de 1 a N, y la variable *k* como natural. Derivar instrucciones que cumplan la siguiente especificación:

$\{\text{Pre: } k = K \wedge s = \Sigma\alpha\text{:}1 \leq \alpha \leq k\text{: } a(\alpha)\}$
$\{\text{Post: } k = K + 1 \wedge s = \Sigma\alpha\text{:}1 \leq \alpha \leq k\text{: } a(\alpha)\}$

Establecer las condiciones de dominio sobre *k* necesarias para que el programa sea válido.

5. Precisar las diferencias entre los tres fragmentos de programa siguientes:

{A1} I {A2}
{A1} [B --> I] {A2}
{A1} [B --> I
    ¬B --> seguir
    ] {A2}

¿Cuál de ellos corresponde a la instrucción C++ "if (B) I" ?

6. Derivar y verificar una instrucción alternativa que, sin modificar el valor de la variable h, cumpla la siguiente especificación:

$\{\text{Pre: } h = N\alpha\text{: } 1 \leq \alpha \leq k\text{: } a(\alpha) = x\}$
$\{\text{Post: } h = N\alpha\text{: } 1 \leq \alpha \leq k + 1\text{: } a(\alpha) = x\}$

¿Qué condiciones de dominio es preciso incorporar a la precondición?

7. Suponiendo que todas las estructuras usadas tienen naturales como elementos, escribir asertos que expresen:

a) que la pila *p* no es nula;
b) que la cima de la pila *p* coincide con el primer elemento de la cola *q*;
c) que la cima de *p* coincide con la raíz del árbol binario *a*;

8. Usando la especificación algebraica del tipo vector de enteros, identificar el carácter de cada uno de los accesos al vector *a* en las siguientes asignaciones y postcondiciones, y calcular las precondiciones correspondientes:

```
 a(a(i)):= j        a(a(i)):=j        a(a(i)):=i
{a(a(i)) = j}      {a(a(j)) = i}     {a(i) = i}
```

9. Los conjuntos de elementos tienen como constructoras la constante que da el conjunto vacío y la operación de añadir un elemento a un conjunto. Otras operaciones útiles son las de sacar un elemento, formar la unión y comprobar si un conjunto es vacío. Especificar el tipo conjunto. Es preciso escribir ecuaciones que relacionen constructoras entre sí. Conviene considerar que sacar un elemento del conjunto vacío no es un error.

10. Usando la operación altura como oculta, establecer las condiciones de dominio necesarias y calcular la precondición correspondiente a:

```
{Pre: ?}
      p:= desapilar(p);
{Post: h = altura(p)}
```

11. En las mismas condiciones que el ejercicio anterior, completar el siguiente fragmento de programa demostrando su corrección:

```
{Pre: h = H ∧ h > 0 ∧ h = altura(p)}
      ?
{Post: h = H - 1 ∧ h = altura(p)}
```

12. Escribir las ecuaciones de la operación *suma*, que suma todos los elementos de una cola de naturales, completando las ecuaciones:

```
suma(c_nula) ≡ ?
suma(pide_turno(n,c)) ≡ ?
```

donde *c* es cualquier cola de naturales y *n* cualquier natural. Considerando ocultas *suma* y *long*, completar el siguiente fragmento de programa, modificando sólo las variables *s* y *c*; demostrar su corrección:

```
{Pre: suma(c) = s ∧ long(c) = L}
      ?
{A1: suma(c) – x = s ∧ long(c) = L + 1 }
      ?
{Post: suma(c) = s ∧ long(c) = L + 1}
```