FUNDAMENTOS DE COMPUTACIÓN
PRÁCTICO 3
NATURALES

Ej. 1 (a) Defina la función `pos::N -> Bool` que recibe un natural y decide si es positivo `(> 0)` o no.

(b) Defina la función `par::N -> Bool` que recibe un natural y decide si es par o no.

(c) Defina la función `impar::N -> Bool` que recibe un natural y decide si es impar o no, sin usar la función `par`.

(d) Defina la función `doble::N -> N` que recibe un natural calcula su doble (sin funciones auxiliares).

(e) Defina la función `triple::N -> N` que recibe un natural calcula su triple (sin funciones auxiliares).

(f) Demuestre que `(∀n::N) par(doble n) = True`.

(g) Demuestre que `(∀n::N) par n = not (impar n)`, con la primera definición de `impar`.

(h) Demuestre que `(∀n::N) n ≤ doble n`.

(i) Demuestre que `(∀n::N) doble n ≤ triple n`.

Puede utilizar los lemas de $\leq$ vistos en clase.

Ej. 2 (a) Defina la función `todos::N -> (N -> Bool) -> Bool`, que recibe un natural `n` y un predicado `p`, y devuelve `True` para todos los números entre `0` y `n` se cumple que `p` es verdadero.
Ejemplos: `todos tres par = False`
`todos cuatro (< cinco) = True`

(b) Defina la función `contar::N -> (N -> Bool) -> N`, que recibe un natural `n` y un predicado `p`, y computa la cantidad de números entre `0` y `n` para los cuales `p` es verdadero.
Ejemplo: `contar par (S(S(S 0))) = S (S 0)`

(c) Demuestre que `(∀n::N) contar n pos = n`.

(d) Demuestre que: $(\forall$`n::N`,$\forall$`p::N -> Bool) contar n p` $\leq$ `S n`.
Puede utilizar los lemas de $\leq$ vistos en clase.

Ej. 3 (a) Defina la función `(==)::N -> N -> Bool`, que recibe dos naturales y devuelve `True` si ambos son iguales.

(b) Defina la función `(<)::N -> N -> Bool`, que recibe dos naturales y devuelve `True` si el primero es menor que el segundo.

(c) Demuestre que `(==)` conmutativa, o sea: $(\forall$`m::N`,$\forall$`n::N) m == n = n == m`.

Ej. 4 (a) Defina la función `(+)::N -> N -> N`, que calcula la suma de dos naturales.

(b) Demuestre que $(\forall$`n::N) doble n = n + n`.

(c) Demuestre que $(\forall$`m::N`,$\forall$`n::N) par (m+n) = par m == par n`.

(d) Demuestre que la suma es asociativa y conmutativa.

Ej. 5 (a) Defina la función `(*)::N -> N -> N`, que calcula el producto de dos naturales.

(b) Demuestre que el producto es conmutativo.

(c) Demuestre que el producto distribuye sobre la suma, o sea:
$(\forall$`m::N`,$\forall$`n::N`,$\forall$`k::N) (m + n) * k = (m * k) + (n * k)` .

(d) Demuestre que el producto es asociativo.

Ej. 6 (a) Defina las función potencia `(^)::N -> N -> N`, que recibe un natural `m` y un natural `n` y devuelve el resultado de elevar `m` a la `n` (o sea, $\mathtt{m}^{\mathtt{n}}$).

(b) Demuestre que $(\forall$`m::N`,$\forall$`n::N`,$\forall$`k::N) m^(n+k) = m^n * m^k` .

(c) Demuestre que $(\forall$`m::N`,$\forall$`n::N`,$\forall$`k::N) (m^n)^k = m^(n*k)` .

Ej. 7 (a) Defina la función `sumi::N -> N`, que recibe un natural `n` y calcula la sumatoria de todos los naturales menores o iguales que `n` (o sea, $\sum_{i=0}^{\mathtt{n}} i$).

(b) Defina la función `sumfacts::N -> N`, que recibe un natural `n` y calcula la sumatoria de los factoriales de todos los naturales menores o iguales que `n` (o sea, $\sum_{i=0}^{\mathtt{n}} i!$).

(c) Defina la función `sumfi::(N->N)->N->N`, que recibe un natural `n` y una función `f` y computa la sumatoria de `(f i)` para `i = 0,...n` ($\sum_{i=0}^{\mathtt{n}}(f\ i)$).

(d) Reescriba la función `sumfacts` utilizando `sumfi` (un renglón).

(e) Defina la función `sumpi::(N->Bool)->N->N`, que recibe un natural `n` y un predicado `p` y computa la sumatoria de todos los naturales menores o iguales que `n` para los cuales `p` es verdadero.

(f) Demuestre que (∀`n::N`,∀`p::N->Bool`) `sumpi n p` $\leq$ `sumi n p`.

Puede utilizar los lemas de $\leq$ vistos en clase.

Ej. 8 (a) Defina la función `min::N -> N -> N` que calcula el mínimo de dos números naturales.

(b) Defina la función `max::N -> N -> N` que calcula el máximo de dos números naturales.

(c) Demuestre que (∀`n::N`) `min n n = n`.

(d) Demuestre que (∀`m::N`,∀`n::N`) `max m n = max n m`.

(e) Demuestre que (∀`m::N`,∀`n::N`) `m` $\leq$ `max n m`.

(f) Demuestre que (∀`m::N`,∀`n::N`) `min m n` $\leq$ `max m n`.

(g) Demuestre que (∀`m::N`,∀`n::N`) `min m (m+n) = m`.

Puede utilizar los lemas de $\leq$ vistos en clase.