3.2 Inducción matemática

Nota 3.2.1 (Principio de inducción matemática)**.** Para demostrar una propiedad P para todos los números naturales basta probar que el 0 tiene la propiedad P y que si n tiene la propiedad P, entonces $n+1$ también la tiene.

$$\frac{P\ 0 \qquad \bigwedge nat.\ \dfrac{P\ nat}{P\ (Suc\ nat)}}{P\ nat}$$

Nota 3.2.2 (Ejemplo de demostración por inducción)**.** Usaremos el principio de inducción matemática para demostrar que

$$1+3+\cdots+(2n-1)=n^2$$

Definición 3.2.3 (Suma de los primeros impares)**.** *suma-impares n es la suma de los n primeros números impares.*

primrec *suma-impares* :: *nat* ⇒ *nat* **where**
suma-impares 0 = 0
| *suma-impares (Suc n) = (2 ∗ (Suc n) − 1) + suma-impares n*

Lema 3.2.4 (Ejemplo de suma de impares)**.** *La suma de los 3 primeros números impares es 9.*

lemma *suma-impares 3 = 9*
by (*simp add:suma-impares-def*)

Nota 3.2.5. La suma de los 3 primero número impares se puede calcular mediante

value *suma-impares 3*

que devuelve el valor *9*

Lema 3.2.6 (Ejemplo de demostración por inducción matemática)**.** *La suma de los n primeros números impares es* n^2.

Nota 3.2.7. Demostración automática del lema 3.2.6.

lemma *suma-impares n = n ∗ n*
by (*induct n*) *simp-all*

Nota 3.2.8 (Los métodos *induct* y *simp_all*)**.** En la demostración **by** *(induct n) simp_all* se aplica inducción en n y los dos casos se prueban por simplificación.

Nota 3.2.9. Demostración con patrones del lema 3.2.6.

lemma *suma-impares* $n = n * n$ (**is** *?P n*)
proof (*induct n*)
 show *?P 0* **by** *simp*
next
 fix *n* **assume** *?P n*
 thus *?P*(*Suc n*) **by** *simp*
qed

Nota 3.2.10 (Patrones)**.** Cualquier fórmula seguida de (**is** *patrón*) equipara el patrón con la fórmula.

Nota 3.2.11. Demostración con patrones y razonamiento ecuacional del lema 3.2.6.

lemma *suma-impares* $n = n * n$ (**is** *?P n*)
proof (*induct n*)
 show *?P 0* **by** *simp*
next
 fix *n* **assume** *HI*: *?P n*
 have *suma-impares* $(Suc\ n) = (2 * (Suc\ n) - 1) +$ *suma-impares n* **by** *simp*
 also have $\ldots = (2 * (Suc\ n) - 1) + n * n$ **using** *HI* **by** *simp*
 also have $\ldots = n * n + 2 * n + 1$ **by** *simp*
 finally show *?P*(*Suc n*) **by** *simp*
qed

Nota 3.2.12. Demostración por inducción y razonamiento ecuacional del lema 3.2.6.

lemma *suma-impares* $n = n * n$
proof (*induct n*)
 show *suma-impares* $0 = 0 * 0$ **by** *simp*
next
 fix *n* **assume** *HI*: *suma-impares* $n = n * n$
 have *suma-impares* $(Suc\ n) = (2 * (Suc\ n) - 1) +$ *suma-impares n* **by** *simp*
 also have $\ldots = (2 * (Suc\ n) - 1) + n * n$ **using** *HI* **by** *simp*
 also have $\ldots = n * n + 2 * n + 1$ **by** *simp*
 finally show *suma-impares* $(Suc\ n) = (Suc\ n) * (Suc\ n)$ **by** *simp*
qed

Definición 3.2.13 (Números pares)**.** *Un número natural* n *es par si existe un natural* m *tal que* $n = m + m$.

definition *par* :: *nat* ⇒ *bool* **where**
 par n ≡ ∃ *m*. *n*=*m*+*m*

Lema 3.2.14 (Ejemplo de inducción y existenciales)**.** *Para todo número natural n, se verifica que n*∗(n+1) es par.*

lemma
 fixes *n* :: *nat*
 shows *par* (*n*∗(*n*+1))
proof (*induct n*)
 show *par* (0 ∗ (0 + 1)) **by** (*simp add:par-def*)
next
 fix *n* **assume** *par* (*n*∗(*n*+1))
 hence ∃ *m*. *n*∗(*n*+1) = *m*+*m* **by** (*simp add:par-def*)
 then obtain *m* **where** *m*: *n*∗(*n*+1) = *m*+*m* **by** (*rule exE*)
 hence (*Suc n*)∗((*Suc n*)+1) = (*m*+*n*+1)+(*m*+*n*+1) **by** *auto*
 hence ∃ *m*. (*Suc n*)∗((*Suc n*)+1) = *m*+*m* **by** (*rule exI*)
 thus *par* ((*Suc n*)∗((*Suc n*)+1)) **by** (*simp add:par-def*)
qed

En Isabelle puede demostrarse de manera más simple un lema equivalente usando en lugar de la función *par* la función predefinida *even*.

lemma
 fixes *n* :: *nat*
 shows *even* (*n*∗(*n*+1))
by *auto*

Para completar la demostración basta deostrar la equivalencia de las funciones *par* y *even*.

lemma
 fixes *n* :: *nat*
 shows *par n* = *even n*
proof −
 have *par n* = (∃ *m*. *n* = *m*+*m*) **by** (*simp add:par-def*)
 thus *par n* = *even n* **by** *presburger*
qed

En la demostración anterior hemos usado la táctica *presburger* que corresponde a la aritmética de Presburger.