

# R11.thy

## De EjerciciosLMF2014

```
header {* R2: Razonamiento sobre programas en Isabelle/HOL *}

theory R2
imports Main
begin

text {* -----
Ejercicio 1. Definir la función
  sumaImpares :: nat ⇒ nat
tal que (sumaImpares n) es la suma de los n primeros números
impares. Por ejemplo,
  sumaImpares 5 = 25
----- *}

fun sumaImpares :: "nat ⇒ nat" where
  "sumaImpares n = undefined"

value "sumaImpares 5" -- "= 25"

text {* -----
Ejercicio 2. Demostrar que
  sumaImpares n = n*n
----- *}

lemma "sumaImpares n = n*n"
oops

text {* -----
Ejercicio 3. Definir la función
  sumaPotenciasDeDosMasUno :: nat ⇒ nat
tal que
  (sumaPotenciasDeDosMasUno n) = 1 + 2^0 + 2^1 + 2^2 + ... + 2^n.
Por ejemplo,
  sumaPotenciasDeDosMasUno 3 = 16
----- *}

fun sumaPotenciasDeDosMasUno :: "nat ⇒ nat" where
  "sumaPotenciasDeDosMasUno n = undefined"

value "sumaPotenciasDeDosMasUno 3" -- "= 16"

text {* -----
Ejercicio 4. Demostrar que
  sumaPotenciasDeDosMasUno n = 2^(n+1)
----- *}

... Por ejemplo,
copia 3 x = [x,x,x]
----- *}

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.
Si la información contenida en el documento es ilícita o lesiona derechos de un tercero háganoslo saber y será retirada.
```



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

```
"copia n x = undefined"
```

```
value "copia 3 x" -- "= [x,x,x]"
```

```
text {* -----  
Ejercicio 6. Definir la función  
  todos :: ('a ⇒ bool) ⇒ 'a list ⇒ bool  
tal que (todos p xs) se verifica si todos los elementos de xs cumplen  
la propiedad p. Por ejemplo,  
  todos (λx. x>(1::nat)) [2,6,4] = True  
  todos (λx. x>(2::nat)) [2,6,4] = False  
Nota: La conjunción se representa por ^  
----- *}
```

```
fun todos :: "('a ⇒ bool) ⇒ 'a list ⇒ bool" where  
"todos p xs = undefined"
```

```
value "todos (λx. x>(1::nat)) [2,6,4]" -- "= True"  
value "todos (λx. x>(2::nat)) [2,6,4]" -- "= False"
```

```
text {* -----  
Ejercicio 7. Demostrar que todos los elementos de (copia n x) son  
iguales a x.  
----- *}
```

```
lemma "todos (λy. y=x) (copia n x)"  
oops
```

```
text {* -----  
Ejercicio 8. Definir la función  
  factR :: nat ⇒ nat  
tal que (factR n) es el factorial de n. Por ejemplo,  
  factR 4 = 24  
----- *}
```

```
fun factR :: "nat ⇒ nat" where  
"factR n = undefined"
```

```
value "factR 4" -- "= 24"
```

```
text {* -----  
Ejercicio 9. Se considera la siguiente definición iterativa de la  
función factorial  
  factI :: "nat ⇒ nat" where  
  factI n = factI' n 1  
  
  factI' :: nat ⇒ nat ⇒ nat" where  
  factI' 0      x = x  
  factI' (Suc n) x = factI' n (Suc n)*x  
Demostrar que, para todo n y todo x, se tiene  
  factI' n x = x * factR n
```

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

```
value "factI 4" -- "= 24"
```

```
lemma "factI n x = x * factR n"  
oops
```

www.cartagena99.com no se hace responsable de la información contenida en el presente documento en virtud al  
Artículo 17.1 de la Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico, de 11 de julio de 2002.

Si la información contenida en el documento es ilícita o lesiona bienes o derechos de un tercero háganoslo saber y será retirada.

```
text {* -----
Ejercicio 10. Demostrar que
factI n = factR n
----- *}
```

```
corollary "factI n = factR n"
oops
```

```
text {* -----
Ejercicio 11. Definir, recursivamente y sin usar (@), la función
amplia :: 'a list => 'a => 'a list
tal que (amplia xs y) es la lista obtenida añadiendo el elemento y al
final de la lista xs. Por ejemplo,
amplia [d,a] t = [d,a,t]
----- *}
```

```
fun amplia :: "'a list => 'a => 'a list" where
"amplia xs y = undefined"
```

```
value "amplia [d,a] t" -- "= [d,a,t]"
```

```
text {* -----
Ejercicio 12. Demostrar que
amplia xs y = xs @ [y]
----- *}
```

```
lemma "amplia xs y = xs @ [y]"
```

```
oops
```

```
end
```

Obtenido de "<http://www.glc.us.es/~jalonso/ejerciciosLMF2014/index.php5/R11.thy>"

The logo for Cartagena99 features the text 'Cartagena99' in a stylized, blue, serif font. The '99' is significantly larger and more prominent than the 'Cartagena' part. The text is set against a light blue background with a subtle gradient and a soft shadow effect.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70