

Apellidos, Nombre:

DNI:

Examen PED abril 2016

Modalidad 0

Normas:

- Tiempo para efectuar el test: **20 minutos**.
- Una pregunta mal contestada elimina una correcta.
- Las soluciones al examen se dejarán en el campus virtual.
- **Una vez empezado el examen no se puede salir del aula hasta finalizarlo.**
- En la **hoja de contestaciones** el verdadero se corresponderá con la **A**, y el falso con la **B**.

	V	F		
En la especificación algebraica, una operación es una función que toma como parámetros (entrada) uno o más valores de diversos tipos, y produce como resultado un solo valor de otro tipo.	<input type="checkbox"/>	<input type="checkbox"/>	1	F
Las ecuaciones (vistas en clase) que permiten realizar la multiplicación de números naturales son las siguientes: VAR x, y: natural; mult(cero, x) = cero mult(x, cero) = cero mult(suc(y), x) = suma(mult(y, x), x)	<input type="checkbox"/>	<input type="checkbox"/>	2	V
En la especificación algebraica, para el tratamiento de errores se añade una constate a la signatura que modeliza un valor de error, por ejemplo $\text{error}_{\text{nat}} \rightarrow \text{natural}$.	<input type="checkbox"/>	<input type="checkbox"/>	3	V
En C++, si se declara un objeto <i>a</i> (p. ej. <i>TPoro a;</i>) cuando la variable <i>a</i> se sale de ámbito entonces se invoca automáticamente al destructor de ese objeto.	<input type="checkbox"/>	<input type="checkbox"/>	4	V
Las ecuaciones (vistas en clase) para la operación <i>recu</i> de un vector son las siguientes: recu(crear(), i) = error() recu(asig(v, i, x), j) si (i == j) entonces j si no recu(v, j) fsi	<input type="checkbox"/>	<input type="checkbox"/>	5	F
La complejidad temporal de la operación <i>desapilar</i> (vista en clase) utilizando vectores (con un índice que indica la cima de la pila) o utilizando listas enlazadas es la misma.	<input type="checkbox"/>	<input type="checkbox"/>	6	V
La complejidad temporal del siguiente fragmento de código es $O(n^2)$ int i, j, n, sum; for (i = 4; i < n; i++) { for (j = i-3, sum = a[i-4]; j <= i; j++) sum += a[j]; cout << "La suma del subarray " << i-4 << " es " << sum << endl; }	<input type="checkbox"/>	<input type="checkbox"/>	7	F
En las colas circulares enlazadas vistas en clase, las operaciones <i>encolar</i> y <i>desencolar</i> tienen complejidad temporal $\Theta(1)$.	<input type="checkbox"/>	<input type="checkbox"/>	8	V
Las ecuaciones (vistas en clase) para la operación <i>desencolar</i> son las siguientes: desencolar(crear()) = crear() si esvacía(c) entonces desencolar(encolar(c, x)) = crear() si no desencolar(encolar(c, x)) = encolar(desencolar(c), x)	<input type="checkbox"/>	<input type="checkbox"/>	9	V
Es posible reconstruir un único árbol binario de búsqueda a partir de un recorrido en preorden.	<input type="checkbox"/>	<input type="checkbox"/>	10	V
Un camino en un árbol es una secuencia a_1, \dots, a_s de árboles tal que para todo $i \in \{1, \dots, s-1\}$, a_i es subárbol de a_{i+1} .	<input type="checkbox"/>	<input type="checkbox"/>	11	F
A los árboles generales también se les llama árboles multicamino de búsqueda.	<input type="checkbox"/>	<input type="checkbox"/>	12	F
La semántica de la operación <i>quita_hojas</i> que actúa sobre un árbol binario y devuelve el árbol binario original sin sus hojas es la siguiente: VAR i, d: arbin; x: item; quita_hojas(crea_arbin()) = crea_arbin() quita_hojas(enraizar(crea_arbin(), x, crea_arbin())) = enraizar(crea_arbin(), x, crea_arbin()) quita_hojas(enraizar(i, x, d)) = enraizar(quita_hojas(i), x, quita_hojas(d))	<input type="checkbox"/>	<input type="checkbox"/>	13	F
Profundidad de un subárbol es la longitud del único camino desde la raíz a dicho subárbol.	<input type="checkbox"/>	<input type="checkbox"/>	14	V