

Apellidos:

Nombre:

Convocatoria:

DNI:

## Examen PED febrero 2008

### Modalidad 0

- Normas:**
- La entrega del test no corre convocatoria.
  - Tiempo para efectuar el test: **30 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		
La complejidad logarítmica aparece en algoritmos que descartan muchos valores (generalmente la mitad) en un único paso.	<input type="checkbox"/>	<input type="checkbox"/>	1	V
Dada la sintaxis de la función $IC(lista, item) \rightarrow lista$ , que inserta un elemento a la cabeza de la lista pasada como parámetro y $crear() \rightarrow lista$ , que crea una lista vacía. La siguiente secuencia: $IC(IC(IC(crear(), a), b), c)$ , daría como resultado una lista con los elementos en este orden: $a \rightarrow b \rightarrow c$ , donde $a$ es el primer elemento de la lista	<input type="checkbox"/>	<input type="checkbox"/>	2	F
Dentro de la especificación algebraica de los números naturales definimos la sintaxis de la función $F$ como: $F: natural \rightarrow BOOL$ , y su semántica como: $F(cero)=TRUE$ , $F(suc(cero))=FALSE$ , $F(suc(suc(x)))=F(x)$ . Para el número natural $x=35$ , la función $F$ devolvería $TRUE$ .	<input type="checkbox"/>	<input type="checkbox"/>	3	F
En C++, si un objeto se sale de ámbito entonces se invoca automáticamente al destructor de ese objeto.	<input type="checkbox"/>	<input type="checkbox"/>	4	V
En C++, el constructor de copia recibe como argumento un objeto del mismo tipo pasado por referencia o por valor.	<input type="checkbox"/>	<input type="checkbox"/>	5	F
El algoritmo de búsqueda binaria estudiado en clase (búsqueda de un elemento en un vector ordenado) tiene una complejidad de $\Omega(1)$ .	<input type="checkbox"/>	<input type="checkbox"/>	6	V
La complejidad espacial es la cantidad de recursos espaciales que un algoritmo consume o necesita para su ejecución	<input type="checkbox"/>	<input type="checkbox"/>	7	V
La operación <code>BorrarItem</code> , que borra todas las ocurrencias del item $i$ que se encuentren en la lista, tiene la siguiente sintaxis y semántica: BorrarItem: LISTA, ITEM $\rightarrow$ LISTA BorrarItem( Crear, $i$ ) = Crear BorrarItem( $IC(L1, j)$ , $i$ ) = si ( $i == j$ ) entonces BorrarItem ( $L1, i$ ) sino $IC( BorrarItem (L1, i), j)$	<input type="checkbox"/>	<input type="checkbox"/>	8	V
La semántica de la operación obtener en una lista con acceso por posición es la siguiente ( $IC=InsertarCabeza(Lista, \acute{I}tem)$ , $p$ : posición, $l1$ : lista, $x$ : ítem): obtener( $crear()$ , $p$ )=error_item() si $p == primera(IC(l1, x))$ entonces obtener( $IC(l1, x)$ , $p$ )= $x$ sino obtener( $IC(l1, x)$ , $p$ )= $IC(obtener(l1, p), x)$	<input type="checkbox"/>	<input type="checkbox"/>	9	F
El máximo número de nodos en un nivel $i-1$ de un árbol binario es $2^{i-2}$ , $i \geq 2$	<input type="checkbox"/>	<input type="checkbox"/>	10	V
Sea el TIPO <code>arbin</code> definido en clase. La semántica de la operación nodos es la siguiente: Var $i, d: arbin$ ; $x: item$ ; nodos( $crear\_arbin()$ )=0 nodos( $enraizar(i, x, d)$ )=nodos( $i$ )+nodos( $d$ )	<input type="checkbox"/>	<input type="checkbox"/>	11	F
El coste temporal en su peor caso de insertar una etiqueta en un árbol binario de búsqueda es lineal con la altura del árbol	<input type="checkbox"/>	<input type="checkbox"/>	12	V
Se puede obtener un único árbol 2-3-4 a partir de su recorrido por niveles	<input type="checkbox"/>	<input type="checkbox"/>	13	V

Apellidos:

Nombre:

Convocatoria:

DNI:

## Examen PED febrero 2008

### Modalidad 1

- Normas:**
- La entrega del test no corre convocatoria.
  - \* Tiempo para efectuar el test: **30 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		
El algoritmo de búsqueda binaria estudiado en clase (búsqueda de un elemento en un vector ordenado) tiene una complejidad de $\Omega(1)$ .	<input type="checkbox"/>	<input type="checkbox"/>	1	V
El coste temporal en su peor caso de insertar una etiqueta en un árbol binario de búsqueda es lineal con la altura del árbol	<input type="checkbox"/>	<input type="checkbox"/>	2	V
Dada la sintaxis de la función $IC(lista, item) \rightarrow lista$ , que inserta un elemento a la cabeza de la lista pasada como parámetro y $crear() \rightarrow lista$ , que crea una lista vacía. La siguiente secuencia: $IC(IC(IC(crear(), a), b), c)$ , daría como resultado una lista con los elementos en este orden: $a \rightarrow b \rightarrow c$ , donde $a$ es el primer elemento de la lista	<input type="checkbox"/>	<input type="checkbox"/>	3	F
Dentro de la especificación algebraica de los números naturales definimos la sintaxis de la función $F$ como: $F: natural \rightarrow BOOL$ , y su semántica como: $F(cero)=TRUE$ , $F(suc(cero))=FALSE$ , $F(suc(suc(x)))=F(x)$ . Para el número natural $x=35$ , la función $F$ devolvería $TRUE$ .	<input type="checkbox"/>	<input type="checkbox"/>	4	F
El máximo número de nodos en un nivel $i-1$ de un árbol binario es $2^{i-2}$ , $i \geq 2$	<input type="checkbox"/>	<input type="checkbox"/>	5	V
En C++, el constructor de copia recibe como argumento un objeto del mismo tipo pasado por referencia o por valor.	<input type="checkbox"/>	<input type="checkbox"/>	6	F
En C++, si un objeto se sale de ámbito entonces se invoca automáticamente al destructor de ese objeto.	<input type="checkbox"/>	<input type="checkbox"/>	7	V
La operación <code>BorrarItem</code> , que borra todas las ocurrencias del item $i$ que se encuentren en la lista, tiene la siguiente sintaxis y semántica: BorrarItem: LISTA, ITEM $\rightarrow$ LISTA BorrarItem( Crear, i) = Crear BorrarItem( IC(L1, j), i) = si ( $i == j$ ) entonces BorrarItem (L1, i ) sino IC ( BorrarItem (L1, i ), j )	<input type="checkbox"/>	<input type="checkbox"/>	8	V
La semántica de la operación obtener en una lista con acceso por posición es la siguiente (IC= InsertarCabeza(Lista, Ítem), p: posición, l1: lista, x: ítem): obtener(crear(),p)=error_item() si $p == primera(IC(l1, x))$ entonces obtener(IC(l1, x), p)=x sino obtener(IC(l1, x), p)=IC(obtener(l1, p), x)	<input type="checkbox"/>	<input type="checkbox"/>	9	F
La complejidad espacial es la cantidad de recursos espaciales que un algoritmo consume o necesita para su ejecución	<input type="checkbox"/>	<input type="checkbox"/>	10	V
La complejidad logarítmica aparece en algoritmos que descartan muchos valores (generalmente la mitad) en un único paso.	<input type="checkbox"/>	<input type="checkbox"/>	11	V
Se puede obtener un único árbol 2-3-4 a partir de su recorrido por niveles	<input type="checkbox"/>	<input type="checkbox"/>	12	V
Sea el TIPO arbin definido en clase. La semántica de la operación nodos es la siguiente: Var i,d:arbin; x:item; nodos(crear_arbin())=0 nodos(enraizar(i,x,d))=nodos(i)+nodos(d)	<input type="checkbox"/>	<input type="checkbox"/>	13	F

Apellidos:

Nombre:

Convocatoria:

DNI:

## Examen PED febrero 2008

### Modalidad 2

- Normas:**
- La entrega del test no corre convocatoria.
  - \* Tiempo para efectuar el test: **30 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		
Dentro de la especificación algebraica de los números naturales definimos la sintaxis de la función F como: $F: \text{natural} \rightarrow \text{BOOL}$ , y su semántica como: $F(\text{cero}) = \text{TRUE}$ , $F(\text{suc}(\text{cero})) = \text{FALSE}$ , $F(\text{suc}(\text{suc}(x))) = F(x)$ . Para el número natural $x=35$ , la función F devolvería TRUE.	<input type="checkbox"/>	<input type="checkbox"/>	1	F
La operación <code>BorrarItem</code> , que borra todas las ocurrencias del item i que se encuentren en la lista, tiene la siguiente sintaxis y semántica: BorrarItem: LISTA, ITEM $\rightarrow$ LISTA BorrarItem( Crear, i) = Crear BorrarItem( IC(L1,j), i) = si ( i == j ) entonces BorrarItem ( L1, i ) sino IC ( BorrarItem ( L1, i ), j )	<input type="checkbox"/>	<input type="checkbox"/>	2	V
La complejidad logarítmica aparece en algoritmos que descartan muchos valores (generalmente la mitad) en un único paso.	<input type="checkbox"/>	<input type="checkbox"/>	3	V
La complejidad espacial es la cantidad de recursos espaciales que un algoritmo consume o necesita para su ejecución	<input type="checkbox"/>	<input type="checkbox"/>	4	V
En C++, si un objeto se sale de ámbito entonces se invoca automáticamente al destructor de ese objeto.	<input type="checkbox"/>	<input type="checkbox"/>	5	V
En C++, el constructor de copia recibe como argumento un objeto del mismo tipo pasado por referencia o por valor.	<input type="checkbox"/>	<input type="checkbox"/>	6	F
El máximo número de nodos en un nivel i-1 de un árbol binario es $2^{i-2}$ , $i \geq 2$	<input type="checkbox"/>	<input type="checkbox"/>	7	V
Sea el TIPO arbin definido en clase. La semántica de la operación nodos es la siguiente: Var i,d:arbin; x:item; nodos(crear_arbin())=0 nodos(enraizar(i,x,d))=nodos(i)+nodos(d)	<input type="checkbox"/>	<input type="checkbox"/>	8	F
Se puede obtener un único árbol 2-3-4 a partir de su recorrido por niveles	<input type="checkbox"/>	<input type="checkbox"/>	9	V
El coste temporal en su peor caso de insertar una etiqueta en un árbol binario de búsqueda es lineal con la altura del árbol	<input type="checkbox"/>	<input type="checkbox"/>	10	V
La semántica de la operación obtener en una lista con acceso por posición es la siguiente (IC= InsertarCabeza(Lista, Ítem), p: posición, l1: lista, x: ítem): obtener(crear(),p)=error_item() si p == primera(IC(l1,x)) entonces obtener(IC(l1,x),p)=x sino obtener(IC(l1,x),p)=IC(obtener(l1,p),x)	<input type="checkbox"/>	<input type="checkbox"/>	11	F
El algoritmo de búsqueda binaria estudiado en clase (búsqueda de un elemento en un vector ordenado) tiene una complejidad de $\Omega(1)$ .	<input type="checkbox"/>	<input type="checkbox"/>	12	V
Dada la sintaxis de la función $IC(\text{lista}, \text{item}) \rightarrow \text{lista}$ , que inserta un elemento a la cabeza de la lista pasada como parámetro y $\text{crear}() \rightarrow \text{lista}$ , que crea una lista vacía. La siguiente secuencia: $IC(IC(IC(\text{crear}(),a),b),c)$ , daría como resultado una lista con los elementos en este orden: $a \rightarrow b \rightarrow c$ , donde a es el primer elemento de la lista	<input type="checkbox"/>	<input type="checkbox"/>	13	F