Apellidos:
Nombre:
Convocatoria:
DNI:

Examen PED diciembre 2005
Modalidad 0

Normas:
- La entrega del test no corre convocatoria.
- 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. A continuación comenzará el siguiente ejercicio.**
- **El test vale un 40% de la nota de teoría.**
- En la **hoja de contestaciones** el verdadero se corresponderá con la **A**, y el falso con la **B**.

	V	F		
Las colas también se conocen como listas FIFO.	❑	❑	1.	V
Un árbol binario lleno es también un árbol 2-3-4.	❑	❑	2.	F
Sea un árbol 2-3-4 inicialmente vacío. Tras utilizar las operaciones de inserción y borrado de un árbol 2-3-4 siempre se consigue un árbol binario lleno.	❑	❑	3.	F
Un árbol binario completo es un árbol 2-3-4.	❑	❑	4.	F
Un árbol binario completo de altura h y con 2^h-1 nodos es un árbol 2-3-4.	❑	❑	5.	F
Un árbol Rojo – Negro con todos sus enlaces negros tiene los mismos nodos que el árbol 2-3-4 equivalente.	❑	❑	6.	V
En un árbol B todos los nodos han de tener al menos m/2 hijos o (m-1)/2 claves.	❑	❑	7.	F
En una tabla Hash con dispersión cerrada, las casillas vacías hay que diferenciarlas de las suprimidas para realizar una inserción.	❑	❑	8.	F
En un Hash cerrado, el factor de carga (α= n/B) puede ser mayor que 1 cuando n sea mayor que B.	❑	❑	9.	F
En un multigrafo pueden existir infinitas aristas para un numero "n" de vértices.	❑	❑	10.	V
La operación concatena es un enriquecimiento de las operaciones definidas para el tipo cola.	❑	❑	11.	V
Es posible reconstruir un único árbol AVL a partir de un recorrido por niveles.	❑	❑	12.	V
El borrado en un árbol AVL nunca requiere más de una rotación en el camino de búsqueda.	❑	❑	13.	F
En la escala de complejidades, la complejidad cuadrática es menor que la logarítmica.	❑	❑	14.	F

Examen PED diciembre 2005

Normas:
- Tiempo para efectuar el ejercicio: **2 horas**
- En la cabecera de cada hoja **Y EN ESTE ORDEN** hay que poner: ***Apellidos, Nombre***.
- Se dispone de 20 minutos para abandonar el examen sin que corra convocatoria.
- Las soluciones al examen se dejarán en el campus virtual.
- Se puede escribir el examen con lápiz, siempre que sea legible
- **Todas las preguntas tienen el mismo valor.** Este examen vale el 60% de la nota de teoría.
- **Publicación de notas de exámenes**: 2 de diciembre. **Fecha de revisión de exámenes:** se publicará en el campus virtual.
- **Los alumnos que estén en 5ª o 6ª convocatoria deben indicarlo en la cabecera de todas las hojas**

1. Utilizando las operaciones definidas en clase para la definición del tipo vector definir la sintaxis y la semántica de la operación *inversos* que indica si dos vectores de naturales dados son inversos entre sí, es decir, si un vector es el inverso del otro. Nota: se asume que el vector está creado con 100 componentes y el rango de las mismas es de 1…100.

2. Dada la siguiente declaración de la clase TArbin que representa un árbol binario de búsqueda cuyas etiquetas son enteros y donde no se permiten etiquetas repetidas, escribe el código del constructor por defecto TArbin(), del destructor ~TArbin() y del método Insertar(int), que devuelve cierto si el número se puede insertar y falso en caso contrario. Escribe también el código del constructor y destructor de la clase TNodo. Importante: se tiene que escribir el código de todos los métodos auxiliares que se empleen.

```
class TArbin {
  public:
    TArbin();
    ...
    ~TArbin();
    ...
    bool Insertar(int);
    ...

  private:
    TNodo *n;
};

class TNodo {
  public:
    TNodo();
    ...
    ~TNodo();
    ...

  private:
    int etiqueta;
    TArbin hijoI, hijoD;
};
```

3. Sea un trie (RETRIEVAL) diseñado para almacenar direcciones de memoria en formato hexadecimal inicialmente vacío y con una complejidad temporal optimizada.

a) Expresa la representación de los nodos del trie para este caso concreto y explica su funcionamiento.
b) Inserta en el trie las siguientes direcciones de memoria: 0001, 000B, AB00, AB01, ABCD, A001, AA11, FF00, FFF0, FF3A, DC45, 000, FF.

4. Dado el DEAP representado en forma del siguiente vector, tal y como se ha visto en clase:

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*	*30*
	1	80	4	7	32	53	5	10	9	8	25	31	40	50	20	15	12	19	35	11	29	30	21	17	24	27	36	28	33

a) Realizar el borrado del mínimo y del máximo en este orden.
b) Dado un índice *i* del vector (en el ejemplo de *a)*, *i* tomará valores entre 2 y 30) que está en un nivel *h* de un DEAP con *n* nodos (en el ejemplo de *a)*, *n* tomará el valor de 30), explicar cómo se obtiene:
   - El nodo padre.
   - El nodo hijo izquierdo e hijo derecho.
   - El nodo simétrico suponiendo que *i* está en el montículo máximo.
   - El nodo simétrico suponiendo que *i* está en el montículo mínimo.

Examen PED diciembre 2005. Soluciones

1.

Sintaxis:

inversos: vector, vector --> bool
invAux: vector, vector --> bool

Semántica:

Var v,w: vector; i,j,x,y: natural;

inversos(v,w) = invAux(v,w) && invAux(w,v)
invAux(crear_vector(),w) = VERDADERO
si x == recu(w,100-i+1) entonces invAux(asig(v,i,x),w) = invAux(v,w)
sino invAux(asig(v,i,x),w) = FALSO

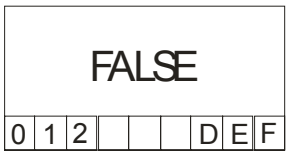
2.

```
TArbin::TArbin() {
  n = NULL;
}

TArbin::~TArbin() {
  if(n != NULL)
  {
    delete n;
    n = NULL;
  }
}

bool
TArbin::Insertar(int item) {
  if(n == NULL)
  {
    n = new TNodo;
    if(n == NULL)
      return false;
    n->etiqueta = item;
    return true;
  }
  else
  {
    if(item < n->etiqueta)
      return (n->hijoI).Insertar(item);
    else if(item > n->etiqueta)
      return (n->hijoD).Insertar(item);
    else
      return false;
  }
}


TNodo::TNodo() {        // Se puede dejar en blanco
  etiqueta = 0;
}

TNodo::~TNodo() {       // Se puede dejar en blanco
  etiqueta = 0;
}
```

3. a) El trie tendrá dos tipos de nodos:

- Nodos internos:
  Son aquellos que se encuentran en el interior de la estructura y que contiene una variable booleana y un vector de 16 punteros a nodo cada uno de los cuales servirá para identificar

cada una de los posibles valores hexadecimales (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, D).
La representación de este nodo será la siguiente:

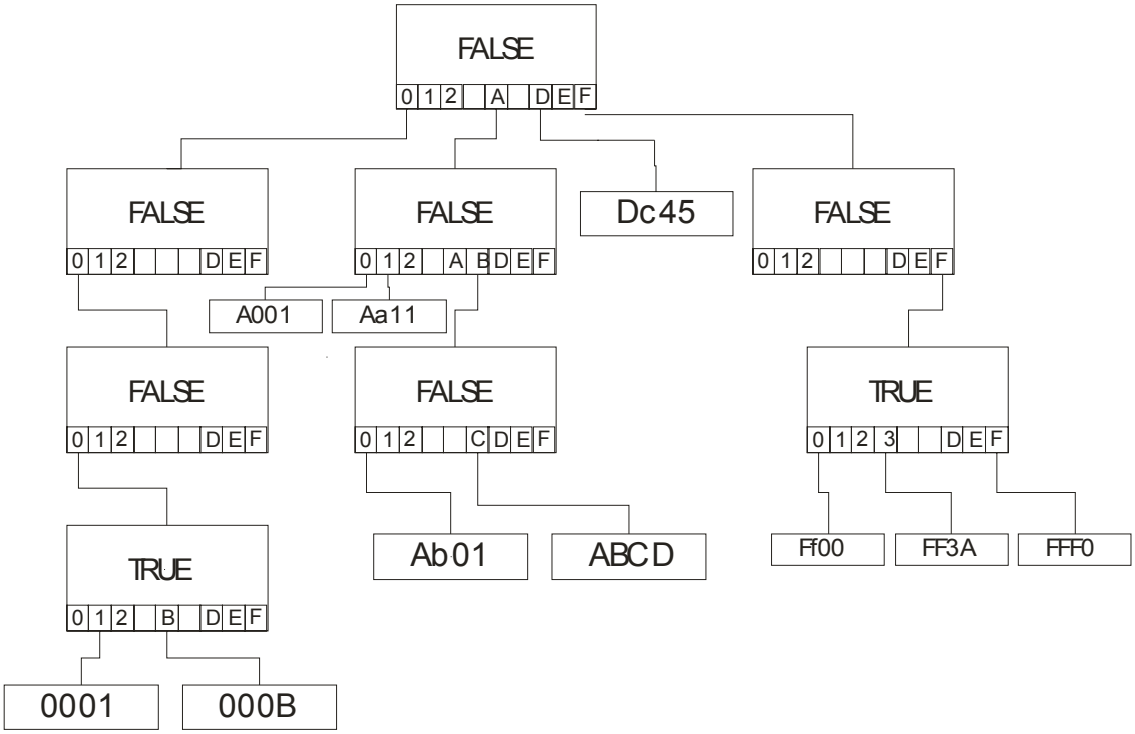


- Nodos terminales:
  Son aquellos que se encuentran en las hojas del Trie y se utilizarán cuando no existan más de una direccion con el mismo prefijo que la que está almacenando.
  La estructura de este nodo será unicamente un array para almacenar la palabra.

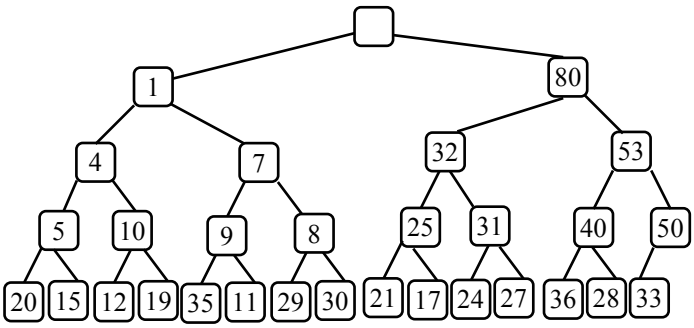
Funcionamiento:

Como se explica en los apuntes.

b)

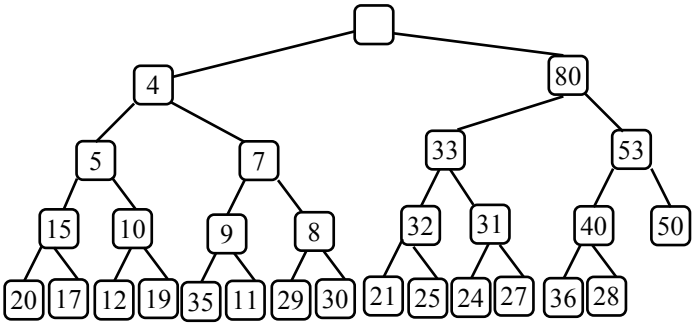


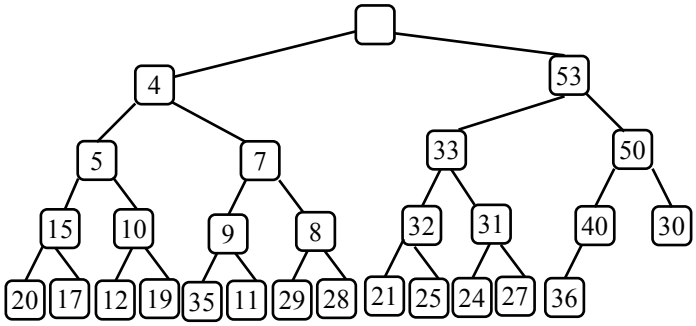
4. a) DEAP de partida, reconstruido a partir del vector:



Borrado del mínimo:

Borrado del máximo:





b)

- El nodo padre: *i DIV 2*
- El nodo hijo izquierdo *2*i* e hijo derecho *2*i+1* (siempre que estos valores sean menores o iguales que *n*, sino no existiría el hijo).
- El nodo simétrico suponiendo que *i* está en el montículo máximo: sabiendo que en el nivel *h* del árbol hay un máximo de 2^{h-1} nodos, habría que restarle la mitad de esos nodos: $i-2^{h-2}$
- El nodo simétrico suponiendo que *i* está en el montículo mínimo: sabiendo que en el nivel *h* del árbol hay un máximo de 2^{h-1} nodos, habría que sumarle la mitad de esos nodos: $i-2^{h-2}$, y si ese valor es mayor que *n* entonces dividirlo por dos para retroceder al padre.