



ASIGNATURA: EDA	 Av. Alcalde de Móstoles 33 (Posterior), 28933, Móstoles (Madrid) 91 664 67 20 - 717 716 540  info@academia-atica.com www.academia-atica.com
TITULACIÓN: Ing Informática	
PROFESORES: Jesús Vergara Igual	
SIMULACRO DE EXAMEN	

Ejercicio 1 [3 puntos]

Pregunta 1

Para almacenar la agenda de teléfonos en un móvil donde no se permite la duplicación del identificador del contacto, ¿qué estructura de datos se debe utilizar para minimizar el tiempo de búsqueda por nombre?

Seleccione una:

- a) Una lista con los datos del contacto como valor.
- b) Un diccionario con los valores del contacto como clave y el nombre del contacto como valor.
- c) Un conjunto con el nombre del contacto como clave.
- d) Dejar sin contestar.
- e) Un diccionario con el nombre del contacto como clave y los datos del contacto como valor.

Pregunta 2

En un árbol B de grado 5, todo nodo no raíz tiene:

Seleccione una:

- a) Un número de hijos entre 2 y 5.
- b) Puede tener cualquier número de hijos menor que 5.
- c) Un número de hijos entre 3 y 5.
- d) Un número de hijos entre 2 y 4.

Pregunta 3

¿Cuál es la cota inferior de complejidad (mejor caso) para encontrar un elemento en una tabla hash?



Selecciona una:

- a) $\Omega(n)$
- b) $\Omega(\log n)$
- c) $\Omega(1)$
- d) Dejar sin contestar
- e) $\Omega(n \log n)$

Pregunta 4

El árbol rojo-negro resultante de insertar sucesivamente la siguiente secuencia de claves [8, 3, 6, 21, 15, 17, 16, 44] da como resultado el siguiente preorden:

Seleccione una:

ASIGNATURA: EDA	 Av. Alcalde de Móstoles 33 (Posterior), 28933, Móstoles (Madrid) 91 664 67 20 - 717 716 540  info@academia-atica.com www.academia-atica.com
TITULACIÓN: Ing Informática	
PROFESORES: Jesús Vergara Igual	
SIMULACRO DE EXAMEN	

- a) 15, 6, 3, 8, 21, 16, 17, 44
- b) Dejar sin contestar
- c) 15, 6, 3, 8, 21, 17, 16, 44
- d) 15, 6, 3, 8, 17, 16, 21, 44
- e) 15, 6, 3, 8, 17, 16, 44, 21

Pregunta 5

Dado el siguiente fragmento de código Java que usa una pila (donde push, pop e isEmpty son las operaciones típicas para inserción, borrado y comprobación de pila vacía, respectivamente), indicar cuál es el resultado impreso.

```
public void prueba () {
    int n=6;
    stack <Integer> p = new Stack <>();
    for (int i=1; i<=n; i++) {
        if (i%2 == 0) {
            p.push(i);
        }
    }
    while (!p.isEmpty()) {
        System.out.print (p.pop());
    }
}
```

Seleccione una:

- a) 246
- b) Dejar sin contestar
- c) 642
- d) 531
- e) 135



Pregunta 6

¿Cómo se conoce el siguiente algoritmo que opera sobre un grafo?

- 1- Se toma el nodo desde el que se inicia la exploración, se etiqueta como explorado y se mete en una cola.
- 2- Se etiquetan como explorados y se insertar en la cola los nodos adyacentes no visitados al de la cabeza de la cola.
- 3- Se extrae el nodo de la cabeza de la cola. Si la cola queda vacía terminar y sino ir al paso 2.

Seleccione una:

- a) Algoritmo de cierre transitivo
- b) Algoritmo de *Dijkstra*
- c) Algoritmo de búsqueda en anchura

ASIGNATURA: EDA	 Av. Alcalde de Móstoles 33 (Posterior), 28933, Móstoles (Madrid) 91 664 67 20 - 717 716 540  info@academia-atica.com www.academia-atica.com
TITULACIÓN: Ing Informática	
PROFESORES: Jesús Vergara Igual	
SIMULACRO DE EXAMEN	

- d) Dejar sin contestar
- e) Algoritmo de búsqueda en profundidad

Pregunta 7

Para deshacer la recursividad de un método con recursividad múltiple se debe utilizar:

Seleccione una:

- a) Dejar sin contestar
- b) Una pila, o una cola si no importa el orden de las llamadas recursivas
- c) Si no importa el orden de las llamadas recursivas se puede usar cualquier estructura de datos dinámica que permita consultar y borrar los elementos que contiene
- d) Una pila solo si no importa el orden de las llamadas recursivas
- e) Una pila, pues es la única estructura de datos que permite deshacer la recursividad.

Pregunta 8

¿Cuál de las siguientes implementaciones corresponde al recorrido en postorden de un árbol?

Seleccione una:

- a)



```

algorithm postOrder(v) {
    for each child w of v{
        postOrder(w)
    }
    visit (v)
}

```

- b) Ninguna es correcta

- c)

ASIGNATURA: EDA	 Av. Alcalde de Móstoles 33 (Posterior), 28933, Móstoles (Madrid) 91 664 67 20 - 717 716 540  info@academia-atica.com www.academia-atica.com
TITULACIÓN: Ing Informática	
PROFESORES: Jesús Vergara Igual	
SIMULACRO DE EXAMEN	

```

algorithm postOrder(v) {
    for each child w of v{
        postOrder(w)
    }
    visit (v)
}

```

d)

```

algorithm postOrder(v) {
    visit (v)
    for each child w of v{
        postOrder(w)
    }
}

```

Pregunta 9

Sea un árbol binario T que contiene 14 nodos. ¿Cuál es el menor valor de la altura posible?

Seleccione una:



- a) 4
- b) Dejar sin contestar
- c) 5
- d) 3
- e) 2

Pregunta 10

Un árbol rojo-negro que contiene en sus nodos n valores enteros tiene como altura un valor máximo de:

Seleccione una:

- a) $2 \log_2 (n+1)$
- b) $\log_2 (n+1)$
- c) Dejar sin contestar
- d) Ninguno de los valores anteriores
- e) $2 \log (n+1)$

ASIGNATURA: EDA	 Av. Alcalde de Móstoles 33 (Posterior), 28933, Móstoles (Madrid) 91 664 67 20 - 717 716 540  info@academia-atica.com www.academia-atica.com
TITULACIÓN: Ing Informática	
PROFESORES: Jesús Vergara Igual	
SIMULACRO DE EXAMEN	

Ejercicio 2 [3.5 puntos]

HashTableMapSC $\langle K, V \rangle$ es una estructura de datos de tipo tabla *hash* donde las colisiones se resuelven mediante el uso de listas enlazadas. En particular, cuando se inserta un elemento en un *HashTableMapSC* se utiliza una función de hash para localizar la posición de la tabla en la que hay que insertar el elemento. En caso de que dos elementos tengan la misma clave dispersada, se insertan en la posición dada por la función hash en una lista enlazada.

El problema de esta estructura es que, en caso de colisiones, la complejidad media en el caso de inserción y de búsqueda es $W(n)$. Por ello, se desea sustituir dichas listas por otra estructura de datos que, en caso de colisión tenga una complejidad media de inserción y de búsqueda de $W(1)$.

Se pide crear una clase *HTMEfficient* con las siguientes características:

- [1 punto]** Crear el constructor y los atributos necesarios en *HTMEfficient* para que se permitan inserciones y borrados con una complejidad media de $W(1)$ en caso de colisión.
- [2 puntos]** Crear los métodos put y get de *HTMEfficient* para que la estructura de datos verifique los requisitos de complejidad.
- [0.5 puntos]** Crear el método remove de *HTMEfficient* para que la estructura de datos verifique los requisitos de complejidad.

Ejercicio 3 [3.5 puntos]

Un árbol se define como un grafo que no tiene ciclos y es conexo. Entendiendo la definición anterior:

- [1.25 punto]** Implementar una función que dado un grafo de tipo *Graph* nos devuelva si es o no conexo.
- [1.25 punto]** Implementar una función que dado un grafo de tipo *Graph* nos devuelva si tiene ciclos o no.
- [1 punto]** Implementar una función que dado un grafo de tipo *Graph* nos devuelva si es un árbol o no.