Facultad Politécnica - Departamento de Informática – Carrera: Ingeniería Informática
Algoritmos y Estructura de Datos III - Primer Examen Parcial – 1er Semestre – Sección TQ/TR
Prof. Cristian Cappo / Prof. Juan Talavera Viernes, *15/04/2016* Puntos: 60

EXAMEN SIN MATERIAL. APAGUE SU CELULAR. La hoja de examen se devuelve.
TENGA A MANO HOJAS DE EXAMEN, CALCULADORA, LAPIZ, BORRADOR Y BOLÍGRAFO. Duración 150 minutos.

Nombre y Apellido: _____Nro. CIC:_____

Tema 1 *(10p)*

Se cuenta con 3 opciones de software para procesar N elementos Se corrieron pruebas y tomaron tiempos de las 3 opciones, pero cada opción fue corrida sobre hardware de distintas velocidades. En la tabla de abajo se muestran los tiempos en segundos tomados para distintos valores de N. La opción 1 corrió en una CPU que procesa 1000 MIPS, la opción 2 en una de 100 MIPS, y la opción 3 en una de 10 MIPS. MIPS = Micro instrucciones por segundo.
Puede asumir que 1 micro instrucción = 1 operación elemental.

N	Velocidad del HW en MIPS		
	Opc 1: 1000	**Opc 2: 100**	**Opc 3: 10**
100	50,02	6,64	35,00
200	200,02	15,29	45,00
300	450,02	24,69	55,00
400	800,02	34,58	65,00
500	1250,02	44,83	75,00

Se pide:

a) Determine cual opción de software es la más óptima. Explique detalladamente como llega a esta conclusión.
b) Determine cual opción de software es la menos óptima. Explique detalladamente como llega a esta conclusión.
c) En base a los datos observados, proponga una O(*f(n)*) y una Ω(*f(n)*) para las 3 opciones.

Tema 2 *(8p)*

a) Demostrar, sin utilizar la fórmula de Stirling, que $log\ (n!) = O(n\ log\ n)$
b) Demostrar que (puede utilizar cualquier método)
b.1) $2^n = O(n!)$
b.2) $n! = \Omega\ (2^n)$

Tema 3 *(14p)*

Marcar la respuesta correcta. NO es necesario justificar. ATIENDA: Si marca incorrectamente es *-1p*

a) ¿Cuál es el tiempo asintótico de este código?

```
for i=1 to n do
  for j=1 to n do
     for k=1 to n do
       if ( i < j ) and ( j < k) and A[i] < A[j] and A[j] < A[k] then c = c + 1;
```

__ $O\ (log\ n)$ __ $O(\ n\ log\ n)$ __ $O(\ n^2)$ __ $O(\ n^3)$

b) Comparado con el pseudocódigo de arriba el siguiente código es

```
for i=n to n do
  for j=i+1 to n do
     for k=j+1 to n do
          if A[i] < A[j] and A[j] < A[k] then c = c + 1;
```

__ Asintóticamente más rápido __ Asintóticamente igual __ Asintóticamente más lento

c) ¿Cuál de las siguientes estructuras de datos para un conjunto *S* debería utilizarse para obtener la más rápida implementación del siguiente código, asumiendo que *S* es inicialmente vacío.?

```
for i = 1 to n do
   if ( buscar(A[i], S) == A[i] ) then remover(A[i],S)
   else insertar(A[i], S)
```

__ Lista doblemente enlazada __Árbol Binario de Búsqueda __Tabla de dispersión __Arreglo abierto ordenado

d) Dado $f(n) = 2\ f(n/2) + 27\ n\ log^2\ n$ y $f(1)=1$, entonces $T(n) =$

__$O(n\ log^3\ n)$ __ $O(n\ log^2\ n)$ __ $O(n^{log\ n}\ log\ n)$ __ $O(\ n^{log_{27}\ n})$

e) El tiempo de ejecución del siguiente código es, asumiendo `insertar(i,S)` y `remover(S)` toma un tiempo *O(log |S|)*

```
for i=1 to n do
    insertar(i2, S)
for i=1 to n do
    remover(S)
```

__ $O(n \log n)$ __ $O(n \log^2 n)$ __ $O(n^2)$ __ $O(n^2 \log n)$

f) Dado el siguiente código, el tiempo de ejecución *T(n)* es igual a:

```
int f ( int n ) {
    if n > 3  then  { return  f ( n -1 ) * f ( n - 3 ) ; }
    else return 3;
}
```

__ $T(n-1) + T(n-3) + O(1)$ __ $T(n-1) * T(n-3) + O(1)$ __ $2\,T(n-1) + O(n-3)$ __ $T(n-1) + O(n)$

g) Dado el siguiente código Java, que produce una cadena a partir de otra, removiendo todas las ocurrencias de un caracter específico. ¿Cuál es el tiempo de ejecución del método considerando `n = in.length()` ?

```
String removerCaracter ( String in, char charToRemove ) {
        String out = "";
        for (int i=0; i < in.length(); i ++ ) {
             char c = in.charAt(i);
             if ( c != charToRemove ) out += c;
        }
}
```

__ $O(\log n)$ __ $O(n)$ __ $O(n \log n)$ __ $O(n^2)$

Tema 4 ***(5p)***

Probar por inducción matemática que para cualquier entero positivo $n \in \{1,2,3,4,5,...\}$ tenemos que $\sum_{i=1}^{n} \frac{1}{i(i+1)} = \frac{n}{n+1}$

Tema 5 ***(8p)***

Complete el siguiente cuadro con el tiempo de ejecución (*en cota O*) en el peor caso para las estructuras de datos y operaciones de diccionario siguientes, considerando ***n*** elementos en la estructura.

Operación en L	Lista simplemente enlazada no ordenada	Lista simplemente enlazada ordenada	Lista doblemente enlazada no ordenada	Lista doblemente enlazada ordenada
`buscar(L, k)`				
`insertar (L, x)`				
`borrar (L,x)`				
`sucesor (L, x)`				
`predecesor (L, x)`				
`minimo (L)`				
`maximo (L)`				

Tema 6 ***(5p)***

Implemente la función de búsqueda en un árbol binario sin hacer uso de recursión ni estructuras auxiliares como pilas y colas. Demuestre el costo asintótico de su algoritmo. Su función recibe como parámetro el nodo raíz y un integer a buscar, y retorna el nodo que contiene el valor buscado.

La clase `Nodo` es la siguiente:

```
public class Nodo {
      public int valor;
      public Nodo izq, der;
}
```

Tema 7 ***(10p)***

Suponiendo que en un árbol AVL cada rotación simple tiene un costo de 4 (cuatro) operaciones elementales. Una rotación doble es igual a 2 (dos) rotaciones simples. Cualquier otra operación sobre un nodo tiene un costo de 1 (una) operación elemental. Por otra parte, sobre un árbol binario de búsqueda (BST), cualquier operación sobre un nodo tiene un costo de 1 (una) operación elemental. Asumiendo que se deben insertar ***N e***lementos al árbol:

a) Explique en que condiciones se daría el peor caso de rendimiento para BST
b) Explique en que condiciones se daría el peor caso de rendimiento para AVL
c) Calcule el costo asintótico en términos de *O* del peor caso para BST dejando detalles del cálculo.
d) Calcule el costo asintótico en términos de *O* del peor caso para AVL dejando detalles del cálculo.