UNIVERSIDAD DE COSTA RICA
FACULTAD DE INGENIERÍA
ESC. DE CS. DE LA COMP. E INFORMÁTICA

CI-1221 Estructuras de Datos y Análisis de Algoritmos
II ciclo 2010, Grupo 03

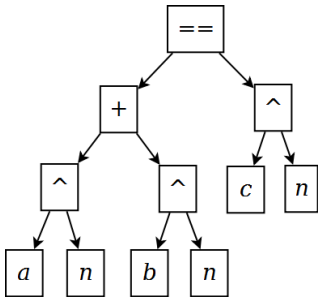
II EXAMEN PARCIAL

El examen consta de 7 preguntas que suman 117 puntos, pero no se reconocerán más de 110 (10 % extra). Cada pregunta empieza con un indicador de su puntaje y del tema tratado. Si la pregunta tiene subítemes, el puntaje de cada uno de ellos es indicado dentro de los subítemes. **No se permite el uso de calculadora**.

1. [17 pts.] *Pilas*

Simule el estado de la pila al ejecutar el siguiente algoritmo sobre la expresión representada por el árbol de la derecha, donde a, b, c y n son enteros y $n > 2$. Los estados de la pila deben ser mostrados en las filas de la matriz adjunta (i-ésima fila $\leftrightarrow$ i-ésimo estado). Los primeros 3 estados son mostrados como cortesía, pero no valen puntos. [1.5 pts. c/ estado, excepto el último: 2.5 pts.].

```
Haga un recorrido del árbol en postorden:
  - Si el nodo es un operando:
    1. Push(op)
  - Si el nodo es un operador:
    1. y = Pop()
    2. x = Pop()
    3. z = x op y
    4. Push(z)
```



a					
a	n				
a^n					

¿A qué famoso teorema corresponde esta expresión? [2.5 pts.]

2. [18 pts.] *Colas*

Una ***cola de doble acceso*** es una cola en la que se permite insertar (extraer) elementos tanto en (desde) la cabeza como en (desde) la cola. Tomando el siguiente código para el encolado y desencolado de elementos en una cola regular (que solo permite insertar en la cola y extraer desde la cabeza), escriba el seudocódigo correspondiente a las operaciones ENQUEUE-HEAD(Q, x), ENQUEUE-TAIL(Q, x), DEQUEUE-HEAD(Q, x) y DEQUEUE-TAIL(Q, x), en una cola de doble acceso. [4 pts. c/ ENQUEUE, 5 pts. c/ DEQUEUE]

Notas. Si lo prefiere:

1. Puede escribir n en vez de $Q.length$.
2. Puede hacer que $Q.tail$ contenga la posición del último elemento en vez de contener la posición de la casilla que le sigue al último elemento.

```
Enqueue(Q,x)
1   Q[Q.tail] = x
2   if Q.tail == Q.length
3       Q.tail = 1
4   else Q.tail = Q.tail + 1
```

```
Dequeue(Q)
1   x = Q[Q.head]
2   if Q.head == Q.length
3       Q.head = 1
4   else Q.head = Q.head + 1
5   return x
```

3. [10 pts.] *Listas enlazadas*

El siguiente es el procedimiento para borrar un elemento de una lista doblemente enlazada con puntero a la cabeza (*head*):

```
List-Delete(L,x)
1   if x.prev <> NIL
2       x.prev.next = x.next
3   else L.head = x.next
4   if x.next <> NIL
5       x.next.prev = x.prev
```

Escriba el código correspondiente al procedimiento de borrado si en vez de guardar un puntero a la cabeza se guardara un puntero a la cola ($L.tail$).

4. [10 pts.] *Tablas hash*

Sea una tabla hash con resolución de colisiones mediante encadenamiento. Utilice la tabla siguiente para indicar la complejidad computacional de búsquedas fallidas y exitosas, en el peor y mejor casos, y en el caso promedio. Asuma que la función hash cumple con la propiedad de *distribución uniforme de los elementos* y que el tamaño de la tabla es $m = O(n)$. [2 pts. c/celda]

Búsqueda\ ***Caso***	**Mejor**	**Promedio**	**Peor**
Fallida			
Exitosa			

5. [10 pts.] *Tablas hash*

Sea una tabla hash de tamaño 10 con resolución de colisiones mediante direccionamiento abierto (*open addressing*) y sondeo exponencial:

$$h(k, i) = (h'(k) + 2^i - 1) \bmod 10.$$

Inserte los elementos 0, 10, 20, 30, 40 y 50 en la tabla y muestre la configuración de la tabla después de cada inserción. Para ello utilice las filas de la siguiente matriz (si lo prefiere puede mostrar en cada fila solo el elemento recién insertado). [2 pts. c/ inserción, + 10 pts. extra el último elemento.]

Elem. insertado\ ***Casilla***	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0										
10										
20										
30										
40										
50										

6. [20 pts.] *Árboles de búsqueda binarios*

a) Inserte en un árbol de búsqueda binario vacío las notas musicales *do, re, mi, fa, sol, la* y *si*, en ese orden, siguiendo el orden lexicográfico y mostrando el estado del árbol después de cada inserción. [2 pts. cada nota. Después de la primera nota mal insertada, el resto de las inserciones no suman puntos].

b) Determine la altura del árbol [2 pts.].

c) Haga un recorrido en orden del árbol, en donde visitar un nodo corresponde a escribir el nombre de la nota. [3 pts.]

d) ¿Qué nota le hace falta a la melodía para que el público se pueda ir contento a su casa? [1 pto.]

7. [32 pts.] *Árboles rojinegros*

a) Inserte en un árbol rojinegro (vacío) las notas musicales *do, re, mi, fa, sol, la* y *si*, en ese orden, siguiendo el orden lexicográfico y mostrando el estado del árbol después de cada inserción [4 pts. cada nota. Después de la primera nota mal insertada, el resto de las inserciones no suman puntos].

b) Determine la altura del árbol y compárela con la altura del árbol de búsqueda binario construido en la pregunta anterior [3 pts.]

c) ¿A qué melodía famosa corresponde un recorrido en orden del árbol [1 pto.]?