UNIVERSIDAD PONTIFICIA DE SALAMANCA EN MADRID
Facultad de Informática
Escuela Universitaria de Informática
Departamento de Lenguajes y Sistemas Informáticos e Ingeniería de Software

Cuadernillo de examen

ASIGNATURA:	**Fundamentos de Programación II**	CÓDIGO:	**113**
CONVOCATORIA:	**Extraordinaria de febrero de 2004**	PLAN DE ESTUDIOS:	**2000**
ESPECIALIDAD:		CURSO:	**1º**
TURNO:	**Tarde**	CURSO ACADÉMICO:	**2003/2004**
CARÁCTER:	**Cuatrimestral (2º cuatrimestre)**	PROGRAMA:	**Ingeniería en Informática / Ingeniería Técnica en Informática**

DURACIÓN APROXIMADA: **2 horas y media**

Soluciones propuestas al examen

Preguntas teórico-prácticas

1. Colas. Concepto de cola. Describa al menos tres formas de implementar colas. Complemente su explicación con un esquema de cómo se haría cada una de esas implementaciones.

Apartado 12.8 del libro de texto

Aplicación
Eligiendo alguna de las tres implementaciones del apartado anterior, codifique un procedimiento que permita insertar un nuevo elemento en una cola. La cola y el elemento a insertar se pasarán como argumentos al procedimiento.

```
procedimiento CInsertar(E/S cola : c  ; E TipoElemento : e)
var
   puntero_a nodo : aux
inicio
   reservar(aux)
   auxE.sig ← nulo
   auxE.info ← e
   si c.p = nulo entonces
      c.p ← aux
   si_no
      c.fE.sig ← aux
   fin_si
   c.f ← aux
fin_procedimiento
```

Puntuación: 1,5 puntos

2. Ordenación de archivos secuenciales. ¿Por qué son diferentes de los métodos de ordenación internos? ¿Qué métodos de ordenación de archivos secuenciales conoce? Describa al menos dos de dichos métodos.

Apartado 11.5 del libro de texto

Aplicación
Codifique un procedimiento que permita realizar una mezcla de dos archivos secuenciales ordenados en secuencias de 4 registros en otro archivo secuencial ordenado en secuencias de 8 registros.

Si los dos archivos de entrada son:

F1:	2	12	15	21	16	18	43	65	10

F2:	1	5	10	22	7	19	26	33

El archivo resultante sería:

F3:	1	2	5	10	15	21	22	7	16	18	19	26	33	43	65	10

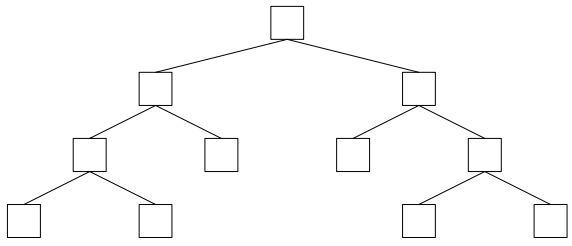
```
procedimiento mezcla(E cadena : NomArch1,NomArch2,NomArch; E entero : n)
//tipoReg es un registro definido que contiene un campo llamado clave
//tipoArch es un archivo secuencial definido con registros de tipo tipoReg
//Para la aplicación planteada se supone que la llamada se hace con n=4
```

```
var
   tipoArch : A,A1,A2
   tipoReg : R1,R2
   entero : i,j
inicio
   crear('ARCH.DAT')
   abrir(A, 'ARCH.DAT',escritura)
   abrir(A1, 'ARCH1.DAT',lectura)
   abrir(A2, 'ARCH2.DAT',lectura)
   leer(A1,R1)
   leer(A2,R2)
   mientras no fda(A1) o no fda(A2) hacer
      i ← 1
      j ← 1
      mientras no fda(A1) y no fda(A2) y (i<=n) y (j<=n) hacer
          si r1.clave < r2.clave entonces
              escribir(A,R1)
              leer(A1,R1)
              i ← i + 1
          si_no
             escribir(A,R2)
             leer(A2,R2)
             j ← j + 1
          fin_si
      fin_mientras
      mientras no fda(A1) y (i<=n) hacer
          escribir(A,R1)
          leer(A1,R1)
          i ← i + 1
      fin_mientras
      mientras no fda(A2) y (j<=n) hacer
          escribir(A,R2)
          leer(A2,R2)
          j ← j + 1
      fin_mientras
   fin_mientras
   cerrar(A,A1,A2)
fin_procedimiento
```

Puntuación: 2 puntos

3. Árboles binarios. Explique el concepto de árbol binario. Explique también los conceptos de árboles similares, árboles idénticos, árbol equilibrado, árbol degenerado. Acompañe su explicación de un dibujo de cada uno de ellos.

Aplicación
Dado el siguiente árbol de expresiones:



Indique que resultados se producen si se recorre el árbol utilizando los recorridos inorden, preorden y postorden.

Recorrido preorden: + * + a b c – d / e f

Recorrido inorden: a + b * c + d – e / f

Recorrido postorden: a b + c * d e f / - +

Puntuación: 1,5 puntos.

Preguntas prácticas

1. Se tiene un archivo secuencial de productos con los siguientes campos:

- Referencia del producto (cadena).
- Descripción del producto (cadena).
- Precio del producto (real)

El archivo ya está ordenado por la referencia del producto y se desea almacenar sus registros en una lista ordenada simplemente enlazada.

Se pide:

a. Describa las estructuras de datos necesarias para almacenar la información (0,5 puntos).

```
tipos
   registro = TipoElemento
      cadena : Ref,Descr
      real : precio
   fin_registro

   puntero_a nodo = lista
   registro = nodo
      TipoElemento : info
      puntero_a nodo = sig
   fin_registro

   archivo_s de TipoElemento = tipoArch
```

b. Codifique un procedimiento que cargue en la lista todos los registros del archivo secuencial (1 punto).

```
procedimiento CargarDatosDeArchivo(S lista : l)
var
   tipoArch : A
   TipoElemento : R
inicio
   abrir(A,'MIARCHIVO.DAT',lectura)
   LeerArchivo(A,l)
   //Leer archivo es un procedimiento recursivo que lee registros del
   //archivo A y los guarda en la lista l
   //Es un procedimiento recursivo porque se desea que los datos queden
   //en la lista en el mismo orden que en el archivo
   cerrar(A)
fin_procedimiento

procedimiento LeerArchivo(E/S tipoArch : A; S lista :l)
var
   TipoElemento : R
inicio
   leer(A,R)
   si no fda(A) entonces
      LeerArchivo(A,l)
      InsertarEnLista(l,R)
   si_no
      l ← nulo
   fin_si
fin_procedimiento

procedimiento InsertarEnLista(E/S lista : l  ; E TipoElemento : e)
var
   lista : aux
inicio
   reservar(aux)
   aux↑.sig ← l
   aux↑.info ← e
   l ← aux
fin_procedimiento
```

c. Codifique un procedimiento que permita modificar el precio de un producto de la lista. La referencia del producto a modificar y el nuevo precio se pasarán como argumentos al procedimiento (1 punto).

```
procedimiento ModificarPrecio(E lista : l  ; E cadena : r; E real: p)
//r es la referencia del artículo a modificar y p el nuevo precio
var
   lista : aux
   lógico : encontrado
inicio
   encontrado ← falso
   aux ← l
   mientras (aux <> nulo) y no encontrado hacer
      si aux↑.info.Ref = r entonces
         encontrado ← verdd
      si_no
         aux ← aux↑.sig
      fin_si
   fin_mientras
   si encontrado entonces
      aux↑.info.precio ← pre
   fin_si
fin_procedimiento
```

d. Escriba un procedimiento que permita eliminar un producto de la lista. La referencia del producto a eliminar se pasará como argumento al procedimiento (1,5 puntos).

```
procedimiento BorrarElemento(E/S lista : l; E cadena r)
var
   lista : aux,ant
   lógico : encontrado
inicio
   encontrado ← falso
   aux ← l
   mientras no EsListaVacía(aux) y no encontrado hacer
      si aux↑.info.Ref = r entonces
         encontrado ← verdd
      si_no
         ant ← aux
         aux ← aux↑.sig
      fin_si
   fin_mientras
   si encontrado entonces
      si aux = l entonces
         Borrar(l)
      si_no
         Borrar(ant↑.sig)
      fin_si
   fin_si
fin_procedimiento

procedimiento Borrar( E/S lista : l)
var
   lista : aux
inicio
   si EsListaVacía(l) entonces
      // error, la lista está vacía
   si_no
      aux ← l
      l ← l↑.sig
      liberar(aux)
   fin_si
fin_procedimiento
```

e. Codifique un procedimiento que vuelque la lista en el archivo secuencial. Al final del mismo, el archivo deberá quedar ordenado por la referencia del producto (1 punto).

```
procedimiento CargarDatosDeLista(E/S lista : l)
var
   tipoArch : A
   TipoElemento : R
   lista : aux
inicio
   crear('MIARCHIVO.DAT')
   abrir(A,'MIARCHIVO.DAT',lectura)
   aux ← l
   mientras no EsListaVacia(aux) hacer
      escribir(A,aux↑.info)
      aux ← aux↑.sig
   fin_mientras
   cerrar(A)
fin_procedimiento
```