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 I**	CÓDIGO:	**106**
CONVOCATORIA:	**Ordinaria de febrero de 2004**	PLAN DE ESTUDIOS:	**2000/2002**
ESPECIALIDAD:		CURSO:	**1º**
TURNO:	**Mañana**	CURSO ACADÉMICO:	**2003/2004**
CARÁCTER:	**Cuatrimestral (1er. cuatrimestre)**	PROGRAMA:	**Ingeniería en Informática / Ingeniería Técnica en Informática**

DURACIÓN APROXIMADA: **2 horas y media**

Soluciones propuestas al examen

Preguntas teórico-prácticas

1. Programación modular. Explique la diferencia entre procedimientos y funciones. Explique la diferencia entre datos locales y globales. Explique como se realiza el paso de parámetros entre el programa principal y los módulos y la diferencia entre el paso de parámetros por valor y por referencia.

Apartados 6.2, 6.3, 6.4 y 6.5 del libro de texto.

Aplicación

Escriba un módulo que permita buscar la posición de un elemento entre una lista de n números enteros. El módulo recibirá como mínimo argumentos para la lista y el elemento a buscar. Escriba dos versiones del módulo, una como procedimiento y otra como función.

```
entero: función Buscar(E vector:v;E TipoElemento:el;E entero:n)
//El tipo vector está definido como un array de 1 a n elementos de tipo entero
var
   entero: i
inicio
   i ← 0
   mientras (el <> v[i]) y (i < n) hacer
      i ← i + 1
   fin_mientras
   si el = v[i] entonces
      devolver(i)
   si_no
      devolver(0)
   fin_si
fin_función

procedimiento Buscar(E vector:v;E TipoElemento:el;E entero:n, E/S entero: pos)
//El tipo vector está definido como un array de 1 a n elementos de tipo entero
var
   entero: i
inicio
   i ← 0
   mientras (el <> v[i]) y (i < n) hacer
      i ← i + 1
   fin_mientras
   si el = v[i] entonces
      pos ← i
   si_no
      pos ← 0
   fin_si
fin_función
```

Puntuación: 1,5 puntos

2. Tipos de dato registro: explique brevemente cuáles son las características del tipo. Enumere las operaciones que se pueden hacer con una variable de este tipo. Escriba la declaración de un tipo registro que contenga los datos: DNI , nombre y apellido, todos ellos de tipo cadena de caracteres.

Apartado 7.7 del libro de texto

Aplicación

Suponemos almacenados en un array n registros como los que ha declarado en el apartado anterior. Escriba un subprograma que reciba dicho array como parámetro y devuelva como resultado los datos (DNI, nombre y apellido) correspondientes al DNI mayor de los almacenados.

```
tipos
   registro = persona
      cadena : dni,nombre, apellidos
   fin_registro
   array[1..20] de persona = personas

cadena función DNIMayor(E personas : p; E entero : n)
var
   entero : i, max
inicio
   max ← 1
   desde i ← 2 hasta n hacer
      si p[i].DNI < p[max].DNI entonces
          max ← i
      fin_si
   fin_desde
   devolver(p[max].DNI
fin_función
```

Puntuación: 1,5 puntos

3. Ordenación interna. Enumere los distintos métodos de ordenación que conozca. Describa el método de ordenación Shell ¿en que se basa su capacidad para ordenar una lista de elementos?

Capítulo 10 (apartado 10.2.4) del libro de texto

Aplicación

Codifique un procedimiento que permita ordenar el array de registros declarado en la segunda pregunta. Deberá emplear el método de ordenación por inserción directa.

```
procedimiento Ordenar(E/S personas:v;E entero : n)
//El vector v tiene elementos entre 0 y n
var
   entero : i,j
inicio
   desde i ← 2 hasta n hacer
      v[0] ← v[i]
      j ← i - 1
      mientras v[i].DNI > v[0].DNI hacer
          v[j+1] ← v[j]
          j ← j - 1
      fin_mientras
      v[j] ← v[0]
fin_desde
fin_procedimiento
```

Puntuación: 2 puntos

Preguntas prácticas

El grupo asegurador MuySeguro está integrado por M compañías de N ramos distintos (automóviles, médicos, hogar, vida etc.). Quiere almacenar en una estructura el número de empleados y el total facturado por cada compañía y ramo, de la manera siguiente:

N Ramos

M Compañías

Número de empleados del ramo	Número de empleados del ramo	Número de empleados del ramo	Número de empleados del ramo	…
Total Facturado	Total Facturado	Total Facturado	Total Facturado	…
Número de empleados del ramo	Número de empleados del ramo	Número de empleados del ramo	Número de empleados del ramo	
Total Facturado	Total Facturado	Total Facturado	Total Facturado	
…				
…				

Nota: Observe que la tabla está compuesta de registros, cada uno de los cuales guarda información sobre el número de empleados del ramo y el total facturado por el ramo, de forma que en la posición 2,3 del array se almacenará el número de empleados y el total facturado de la compañía 2 en el ramo 3.

1. Escriba las declaraciones necesarias para representar dichos datos (0,5 puntos).

```
tipos
   registro = ramo
      entero : NumEmpleados
      real : total
   fin_registro
   array[1..M, 1..N] de ramo = tabla

   //Para el apartado 3 del problema
   array[1..M] de real = TotalPorRamo
   array[1..N de real = TotalPorCompañía
```

2. Escriba un procedimiento que permita leer los datos de un archivo secuencial con los campos `NumEmpleados` (de tipo entero) y `Total` (tipo real) (1,5 puntos)

```
procedimiento CargarDatos(S tabla : t; E entero : n,m)
var
   entero : i,j
   archivo_s de ramo : A
inicio
   abrir(A,'RAMOS.DAT',lectura)
   desde i ← 1 hasta m hacer
      desde j ← 1 hasta n hacer
         leer(A,t[i,j])
      fin_desde
   fin_desde
fin_procedimiento
```

3. Escriba un subprograma que calcule el total facturado por compañía, el total facturado por ramo y el número total de trabajadores del grupo en los n ramos de las m compañías (1,5 puntos)

```
procedimiento Totales(E tabla : t; E entero : m,n ; S TotalPorRamo : TotalRamo;
                                    S TotalPorCompañía : TotalCia; S entero : TotalEmp)
var
   entero : i,j
inicio
   TotalEmp ← 0
   //Calcular el total por ramo y el total de empleados
   desde j ← 1 hasta n hacer
      TotalRamo ← 0
      desde i ← 1 hasta m hacer
         TotalRamo ← TotalRamo + t[i,j].Total
         TotalEmp ← TotalEmp + t[i,j].NumEmpleados
      fin_desde
```

```
   fin_desde

   //Calcular el total por compañía
   desde i ← 1 hasta m hacer
      TotalCia ← 0
      desde j ← 1 hasta n hacer
         TotalCia ← TotalCia + t[i,j].Total
      fin_desde
   fin_desde
fin_procedimiento
```

4. Escriba un módulo que permita averiguar el ramo y la compañía que más ha facturado (1,5 puntos).

```
procedimiento MayorFacturacion(E tabla : t; E entero : m,n; S entero : CMax,RMax)
//CMax y RMax guardarán, respectivamente, la posición
//de la compañía y ramo que más han facturado
var
   entero : i,j
inicio
   CMax ← 1
   RMax ← 1
   desde i ← 1 hasta m hacer
      desde j ← 1 hasta n hacer
         si t[i,j].total > t[CMax,RMax].total
             CMax ← i
             RMax ← j
         fin_si
      fin_desde
   fin_desde
fin_procedimiento
```