C. P. S. - Universidad de Zaragoza

Dpto. Informática e Ingeniería de Sistemas

Intr. a la Programación / Fund. Informática

2ª convocatoria 30 - Agosto - 2005

*Examen teórico (80% de la calificación total)* *Duración total:* 3 horas

## **Ejercicio 1** [3,5 puntos]

Para guardar la representación geométrica de un poliedro se utiliza un fichero de texto que contiene información de los vértices y aristas de la siguiente forma (ver ejemplo):

*Primera línea*: Número de vértices (entero).

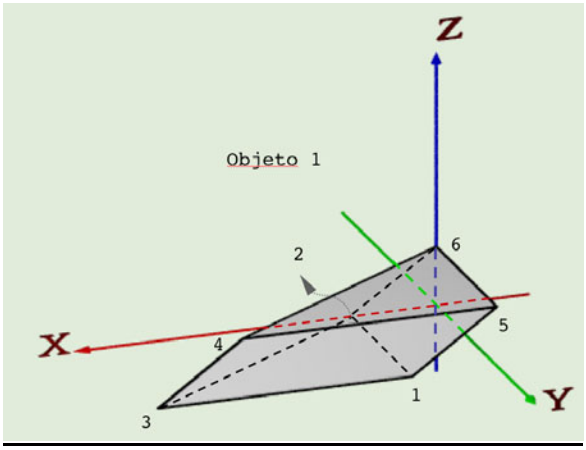
*Siguientes líneas*: Información de cada vértice. En cada línea se indica el identificador del vértice (enteros correlativos, comenzando por 1), seguido de sus coordenadas tridimensionales (números reales). Los datos se separan mediante espacios.

*Siguiente línea*: Número de caras (entero).

*Siguientes líneas*: Información de cada cara. En cada línea se indican los identificadores de los vértices que la definen, ordenados de forma que cada arista se especifica con dos vértices consecutivos; la cara se cierra con una arista que va del último al primero de sus vértices. Por ejemplo, si una cara está formada por los vértices 5 6 4 quiere decir que sus aristas son 5-6, 6-4 y 4-5.

**Ejemplo:**

| Objeto1.txt | Significado |
|---|---|
| 6 | número de vértices |
| 1 1.0 2.0 0.0 | vértice 1 coords. x, y, z |
| 2 1.0 0.0 0.0 | vértice 2 coords. x, y, z |
| 3 4.0 2.0 0.0 | .... |
| 4 3.0 2.0 1.0 | |
| 5 0.0 2.0 1.0 | |
| 6 0.0 0.0 1.0 | vértice 6 coords. x, y, z |
| 5 | número de caras |
| 5 6 4 | vértices cara 1 (3 aristas) |
| 1 3 2 | vértices cara 2 (3 aristas) |
| 4 6 2 3 | vértices cara 3 (4 aristas) |
| 4 3 1 5 | ..... |
| 5 1 2 6 | vértices cara 5 (4 aristas) |




**Tarea:** Desarrollar un programa PASCAL que, a partir del fichero de datos cuyo nombre se pedirá al usuario (en el ejemplo '`objeto1.txt`') calcule la superficie total del objeto. Para ello,

1.- Define las estructuras de datos (`tpVertice`, `tpCara`, `tpObjeto`, etc...) necesarias para almacenar los datos en memoria. Supondremos que no hay objetos de más de 100 caras ni caras de más de 50 vértices.

2.- Desarrolla el procedimiento

```
procedure cargaObjeto(?? miFichero: tpNombreFichero; ?? miObjeto: tpObjeto);
```

*{Carga los datos del fichero de nombre* `miFichero` *en la estructura* `miObjeto` *}*

3.- Utilizando la función predefinida `areaTriang`, que se especifica más abajo, desarrolla la función

```
function areaCara(??? laCara: tpCara):real;
```

*{Devuelve el área de la* `cara` *cuyos datos están almacenados en* `laCara`*}*

4.- Escribe el programa principal que muestre por pantalla el área total (suma del àrea de todas las caras) del poliedro almacenado en el fichero de texto especificado interactivamente por el usuario.

**Notas:** Un polígono puede descomponerse en triángulos tomando un vértice cualquiera y trazando todas las diagonales que parten de él. Con esto, la única expresión matemática necesaria es la del área de un triángulo en función de las coordenadas de sus vértices, que se supone conocida:

```
function areaTriang(v1, v2, v3: tpVertice):real;
```

*{Devuelve el área del triángulo definido por los vértices* `v1, v2, v3`*}*

Se supone que el fichero no contiene errores.

## **Ejercicio 2** [3 puntos]

Sabiendo que un número es divisible por 11 si el valor absoluto de la suma de las cifras en posición par menos la suma de las cifras en posición impar es cero o múltiplo de 11, y que un número es divisible por 5 si su dígito menos significativo es 0 o 5, es posible calcular de un modo simple, y sin utilizar el operador MOD, si un número es divisible por 11 y por 5, incluso para números naturales con un número de cifras tan grande como se quiera.

**Se pide:** escribir un programa en Pascal que solicite un número natural y, haciendo uso de las reglas anteriores, muestre por pantalla si es multiplo de 5 y/o de 11. Debe tenerse en cuenta que el número puede tener tantas cifras como se quiera (es posible que sea mayor que MAXINT y, por tanto, no representable como entero).

**Ejemplo:**

```
Escriba un número natural: 292307565↵
es múltiplo de:  5  11
```

***Sugerencia**:* Sumar y restar las cifras a medida que son leídas, ajustando el resultado a un valor entre 0 y 10 (sumando o restando 11, es decir, módulo 11)

## **Ejercicio 3** [3,5 puntos]

Para la resolución de ciertos crímenes, la policía de Metrópolis cuenta con un sistema informático que compara las huellas dactilares encontradas en la escena del crimen con las de los sospechosos, almacenadas en un fichero de registros. Cada registro representa la ficha de un sospechoso, y contiene su nombre y apellidos, y una pequeña imagen digitalizada de la huella de su pulgar derecho. La imagen está compuesta por pixels (puntos) y codificada en blanco y negro (cada pixel sólo puede tomar dos valores: blanco o negro). El tamaño de cada una de estas imágenes es de 400x200 pixels.

La imagen de una huella hallada en la escena del crimen se codifica de manera idéntica a las de las fichas. Se considera *coincidencia fuera de toda duda* cuando dos huellas coinciden en más del 95% de sus pixels. El programa de la policía compara la imagen de la huella hallada con las del fichero hasta encontrar una que coincide fuera de toda duda, devolviendo los datos del sospechoso asociados a dicha huella. En el caso de no se haya encontrado ninguna, devuelve los datos del sospechoso cuya huella presenta mayor número de pixels coincidentes.

**Se pide:**

a) Diseñar las estructura de datos adecuadas para el problema planteado
b) Escribir el procedimiento *buscarCriminal*, que recibe como parámetros la imagen de una huella hallada en la escena del crimen y el fichero con las fichas de los sospechosos, y muestra por pantalla el nombre y apellidos del sospechoso (bien con *coincidencia fuera de toda duda* o, si no lo hay, el del sospechoso con mayor número de pixels coincidentes), así como el porcentaje de pixels coincidentes.