Examen de **Fundamentos de Informática**
Grado de Tecnologías Industriales
Escuela de Ingeniería y Arquitectura
Segunda convocatoria: 11 de septiembre de 2017

Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

Duración total del examen: ***3 horas***

Nota: Responde cada ejercicio en una hoja diferente. No utilices la misma hoja para dos (o tres) ejercicios.

Ejercicio 1 [3.5 puntos]

Se dice que un número natural es poderoso si, **para todo** divisor suyo *p* que es primo, se cumple que *p* al cuadrado (p^2) también es divisor del número. Por ejemplo, el número 36 es un número poderoso, ya que los únicos primos que son divisores suyos son 2 y 3 y se cumple que 4 (2^2) y 9 (3^2) también son divisores de 36.

Se pide: Desarrollar un programa en Pascal que muestre por pantalla, separados por espacios, los *M* primeros números poderosos, donde *M* es una constante especificada en la sección correspondiente del programa (podéis fijarla a 10 en la resolución). Se valorará el uso adecuado de subprogramas.

Observaciones:

1. Los números primos no son números poderosos.
2. El número 1 sí es un número poderoso.

Ejercicio 2 [3 puntos]

Para el diseño de una nueva placa solar se diseña un experimento aleatorio para cada material probado, y mediante un amperímetro se mide la intensidad de corriente resultante. Los valores de intensidad (en amperios) se guardan en un fichero secuencial de números reales.

Las propiedades interesantes de dicho experimento se analizan a través de sus **momentos estadísticos centrales**, que se calculan a partir de su media *μ*. La media se estima así:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

donde x_i representa cada una de las muestras (valores en amperios) del experimento y *n* representa la cantidad total de muestras tomadas (elementos del fichero). A partir de la misma, se puede calcular el momento central de orden *k* (denominado m_k) con la siguiente fórmula:

$$m_k = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu)^k$$

Se pide: Desarrollar un programa en Pascal que pida al usuario el nombre de un fichero de números reales y un número *k*, y muestre por pantalla el momento central de orden *k* de los datos contenidos en dicho fichero. No existe límite máximo para el número de muestras contenidas en el fichero.

Ejemplos de ejecución:

Contenido del fichero **`amperios.dat`**:

9.781	7.772	8.038	9.005	8.913	9.689	7.985	9.876	7.113	8.471

Ejemplo 1 de ejecución:

```
Introduce el nombre del fichero : amperios.dat
Introduce el orden del momento : 1
El momento de orden 1 es 0.000
```

Ejemplo 2 de ejecución:

```
Introduce el nombre del fichero : amperios.dat
Introduce el orden del momento : 3
El momento de orden 3 es -0.071
```

Ejercicio 3 [3.5 puntos]

La distribución de las teclas en un teclado de un ordenador o un teléfono móvil no es arbitraria, sino que se calcula siguiendo ciertas reglas dependientes del idioma, y por eso puede ser distinta para distintos idiomas. Una de las reglas es que se utilicen, en la mayor medida posible, ambas manos para escribir cada palabra, para que el proceso de escritura sea más rápido. Para ello, se busca que el número de palabras del idioma que se tengan que escribir con una sola mano (porque todas sus letras están en el mismo lado del teclado) sea mínimo. Por tanto, para ver cómo de buena es una distribución de teclado para un idioma, podemos contar cuántas palabras de ese idioma se escriben con una sola mano en esa distribución de teclado (cuanto menor sea ese número, mejor es la distribución de teclado).

Para la realización de este ejercicio supondremos que las palabras sólo están formadas por **letras minúsculas**; no habrá ni mayúsculas, ni números, ni signos de puntuación.

Se pide, en Pascal:

a) Definir una estructura de datos para representar una distribución de teclado (esto es, un tipo de teclado). La estructura debe almacenar, para cada letra (minúscula) del alfabeto, si dicha letra se escribe con la mano izquierda o derecha. Únicamente define la estructura; los valores que contenga dependerán de cada tipo de teclado.

```
type
   tpTeclado = ???
```

b) Dada una palabra representada mediante el tipo de datos *String*, y una definición de un tipo teclado, escribir una función que diga si esa palabra se escribe sólo con una mano (en cuyo caso la función devolverá *true*) o con ambas manos (en cuyo caso devolverá *false*). La cabecera será:

```
function soloUnaMano(??? palabra: String; ??? teclado: tpTeclado): boolean;
```

c) Sea un fichero de texto que contiene un diccionario de palabras, con una palabra en cada línea. Escribe un subprograma que, tomando como parámetros el nombre de dicho fichero y una definición de un tipo de teclado, calcule el número total de palabras que se escriben con una sola mano. La cabecera será:

```
??? palabrasUnaMano(??? nombreFichero: String; ??? teclado: tpTeclado): ???;
```

Observa que, tanto en el apartado `b)` como en el `c)`, se está asumiendo que el parámetro **`teclado`** ya contiene los valores correspondientes a la distribución de teclado en el momento en que se ejecuta el subprograma.

Se valorará el uso correcto de los tipos de datos de Pascal, así como el uso adecuado de subprogramas (procedimientos y/o funciones).

Recuerda que:

1. La función `length` devuelve el número de caracteres (longitud) del dato de tipo *String* suministrado como parámetro.
2. Para acceder a los elementos de un *String* se usa la misma sintaxis que en los vectores.
3. Es posible leer un *String* de un fichero de texto, usando las instrucciones de lectura habituales.