UNIVERSIDAD
DE MÁLAGA

Laboratorio de Programación 1
septiembre 2005
(1º de Ingeniería de Telecomunicación)
E.T.S.I. TELECOMUNICACIÓN

Alumno: ____________ Grupo: ____ Ordenador: ____

1. Según el procedimiento ideado por Blaise Pascal, para comprobar si un número n es múltiplo de k, basta con sumar cada uno de los dígitos de n, previamente multiplicados por unos coeficientes apropiados y calculados en función de k. El proceso se repite con la cifra resultante de dicha suma hasta obtener una cifra de un solo dígito, esto es, un número menor que 10 que, si es múltiplo de k indica que n es múltiplo de k.

   Por ejemplo, si $k = 7$, los coeficientes por los que debemos multiplicar las cifras del número n son: 1, 3, 2, 6, 4 y 5 (luego explicaremos cómo se calculan estos coeficientes). Así, si n es, por ejemplo, el número 21756, para saber si es múltiplo de 7 calculamos la suma de los productos de cada dígito del número 21756 por el coeficiente correspondiente, empezando por el dígito de las unidades, siguiendo por las decenas, etc. Esto es: $6 \times 1 + 5 \times 3 + 7 \times 2 + 1 \times 6 + 2 \times 4 = 6 + 15 + 14 + 6 + 8 = 49$. Ahora, hacemos lo mismo para el número 49 resultante: $9 \times 1 + 4 \times 3 = 9 + 12 = 21$. Y, por último, para el 21 resultante: $1 \times 1 + 2 \times 3 = 1 + 6 = 7$. De esta forma hemos obtenido un número de una sola cifra, que es múltiplo de 7 (y en este caso es, precisamente, 7), por lo que concluímos que el número $n = 21756$ es múltiplo de $k = 7$.

   Para calcular los coeficientes, sea cual sea k ($2 \leq k \leq 9$), empezamos con un 1. El mecanismo general consiste en tomar el último coeficiente calculado, multiplicarlo por 10, y calcular el resto de la división entera por k. Así obtenemos el siguiente coeficiente. Seguimos el proceso hasta que, en alguna iteración, volvemos a obtener algún coeficiente que ya haya salido anteriormente. En ese momento, podemos parar el proceso, dado que la secuencia de coeficientes se repetiría y no tendría sentido seguir.

   De este modo, siguiendo con el ejemplo de $k = 7$, tendríamos los coeficientes:

   1

   $1 \times 10 \ mod\ 7 = 10 \ mod\ 7 = \mathbf{3}$

   $3 \times 10 \ mod\ 7 = 30 \ mod\ 7 = \mathbf{2}$

   $2 \times 10 \ mod\ 7 = 20 \ mod\ 7 = \mathbf{6}$

   $6 \times 10 \ mod\ 7 = 60 \ mod\ 7 = \mathbf{4}$

   $4 \times 10 \ mod\ 7 = 40 \ mod\ 7 = \mathbf{5}$

   $5 \times 10 \ mod\ 7 = 50 \ mod\ 7 = \mathbf{1}$

   En este caso, el proceso se ha parado porque el coeficiente 1 ha vuelto a aparecer. Por tanto los coeficientes para $k = 7$ son 1, 3, 2, 6, 4, 5. Si necesitásemos más coeficientes, porque n tuviese más de 6 dígitos, bastaría con repetir la secuencia de coeficientes hasta completar el número de dígitos de n: 1, 3, 2, 6, 4, 5, 1, 3, 2, 6, ...

   Si, por poner otro ejemplo, $k = 8$, los coeficientes resultantes son 1, 2, 4, 0. No es necesario seguir calculando coeficientes para las siguientes posiciones porque sería, simplemente, seguir repitiendo indefinidamente la secuencia de ceros: 1, 2, 4, 0, 0, 0, 0, ...

   Dato este procedimiento explicado como criterio de divisibilidad de n por k, se pide:

   a) Implementar la función `CalcularCoeficientes` que recibe como parámetro un número natural de una cifra y devuelve, a través de un segundo parámetro por referencia, un vector con los coeficientes calculados según el procedimiento anteriormente expuesto.

b) Implementar la función lógica `EsDivisible` que recibe como parámetro un número natural n con un máximo de 5 dígitos (utilizar para ello el tipo `unsigned long int`) y otro número natural k de un solo dígito. La función debe indicar si el primer número es múltiplo del segundo siguiendo, para ello, el procedimiento anteriormente descrito y haciendo uso de la función `CalcularCoeficientes` implementada en el apartado anterior.

2. Sea una cadena de caracteres que esté compuesta por números naturales y símbolos de función (`+` y `-`) y que represente una expresión aritmética en notación infija como en el siguiente ejemplo:

   ```
   123+7-45-91-20-99+2005
   ```

   Implementar una función tenga como parámetro de entrada una cadena de ese tipo y devuelva el resultado de evaluar la expresión contenida en la cadena.

   Como se ve en el ejemplo, no hay ningún tipo de separadores entre los componentes de la expresión.

Notas sobre el examen

- Se debe trabajar en el directorio `/home/alumno/lp1sep05/`. Si la ruta no existe, deberá crearse.
- El programa principal deberá ser lo más sencillo posible, con el código necesario para probar el funcionamiento de todas las funciones que se implementen.
- Deben entregarse en un disquete etiquetado con el nombre, apellidos, grupo y número de ordenador que contenga los ficheros `pascal.cpp` y `expresion.cpp` con el código fuente del examen.
- El código fuente debe comenzar por un comentario donde se indiquen nombre, apellidos y grupo. Esa información debe mostrarse en pantalla cada vez que se ejecute el código.