Tarea 4: Divide and Conquer

Programación 3
Instituto de Computación
Facultad de Ingeniería de la Universidad de la República
Curso 2010
Versión 1.0

1. Objetivos

- Diseñar e implementar un algoritmo basado en la técnica de Divide and Conquer.

2. Conocimientos previos

- Metodología de programación estructurada.
- Conocimientos básicos de C*.
- Divide and Conquer.

3. Descripción del problema

Desde que se creó *P3 Algorithm*, sus ventas fueron aumentando significativamente año a año y más aún cuando firmó contrato con una multinacional en estos últimos meses.

Como consecuencia el departamento contable se ha visto impedido de realizar sus cálculos habituales, porque las computadoras disponibles no pueden manejar números tan grandes, debido a que los mismos superan la representación interna soportada por las computadoras.

Es aquí donde entra el equipo de desarrollo de la empresa, que debe desarrollar un módulo de operaciones de aritmética básica que permita manejar números muy grandes.

3.1. Módulo Operaciones

El módulo de aritmética básica a desarrollar debe soportar números naturales con una cantidad arbitraria de dígitos en cualquiera de sus operaciones. En general se asumirá que la cantidad de dígitos es potencia de 2 y para aquellas operaciones que involucren más de un número, se asumirá que dichos números tienen la misma cantidad de dígitos.

El módulo debe soportar las siguientes operaciones:

- **multiplicar**: Devuelve un natural con $2 * n$ dígitos, resultado de la multiplicación de dos naturales de n dígitos cada uno. Asumimos que n es una potencia de dos. El método que estamos considerando descompone a y b: $a = 10^{n/2}w + x$ y $b = 10^{n/2}y + z$. Luego, $a * b = 10^n(w * y) + 10^{n/2}(w * z + x * y) + x * z$. Tenga en cuenta que la multiplicación de cualquier número por alguna potencia de 10 puede resolverse sin utilizar la operación definida en este módulo.
- **sumar**: Devuelve un natural de n dígitos, resultado de la suma de dos naturales de n dígitos. Debe indicar si la suma genera acarreo.
- **exponenciar**: Devuelve un natural, resultado de elevar la base por el exponente pasados por parámetro. Asumimos que el exponente es una potencia de dos. Debe indicar la cantiad de dígitos del resultado. Se requiere que este procedimiento se realice en $O(log(E))$ siendo E el exponente.
- **liberarResultado**: Libera la memoria asociada el resultado devuelto por las operaciones mencionadas anteriormente.

En el caso que el usuario quisiera multiplicar o sumar dos números con distinta cantidad de dígitos, primeramente debería rellenar con ceros al número con menos dígitos y luego invocar a la operación correspondiente.

3.2. Representación de los números

La representación elegida para modelar los números son vectores de naturales, donde cada elemento del vector es un dígito del número a representar. Sea $D = d_0d_1d_2 \cdots d_{n-1}$ un número de n dígitos, donde d_0 es el dígito más significativo y d_{n-1} el dígito menos significativo ($0 <= d_i <= 9$). Entonces, d_0 se posiciona en el primer elemento del arreglo y d_{n-1} en el último lugar.

Por ejemplo, la figura 1 muestra la representación del número **2468** en un arreglo de 4 elementos.

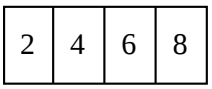

Figura 1: Representación interna del número 2468

3.3. Ejemplo

3.3.1. Código fuente

```
1  /*
2   * Tarea 4: Ejemplo
3   *
4   * Programación 3
5   * Instituto de Computación
6   * Facultad de Ingeniería
7   * Universidad de la República
8   *
9   * $Rev: 289 $
10  */
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include "operaciones.h"
14 #include "utils.h"
15
16 int main() {
17         bool acarreo;
18         int n = 8;
19         int* a = new int[n];
20         int* b = new int[n];
21
22         // numero a
23         a[0] = 0;
24         a[1] = 8;
25         a[2] = 7;
26         a[3] = 6;
27         a[4] = 5;
28         a[5] = 5;
29         a[6] = 4;
30         a[7] = 3;
31
32         // numero b
33         b[0] = 0;
34         b[1] = 0;
35         b[2] = 2;
36         b[3] = 3;
37         b[4] = 4;
38         b[5] = 8;
39         b[6] = 1;
40         b[7] = 9;
41
42         // Multiplico
43         int * c = multiplicar(a, b, n);
44
45         imprimir(a, n); printf(" * "); imprimir(b, n); printf(" = ");
46
47         // imprimo resultado
48         imprimir(c, 2*n);
49         printf("\n");
50         liberarResultado(c);
51
52         // Sumo
53         c = sumar(a, b, n, acarreo);
54
55         imprimir(a, n); printf(" + "); imprimir(b, n); printf(" = ");
56
57         // imprimo resultado
58         if (acarreo) printf("C ");
59         imprimir(c, n); printf("\n");
60         liberarResultado(c);
61
62         delete []a;
63         delete []b;
64         return EXIT_SUCCESS;
65 }
```

3.3.2. Salida

```
8765543 * 234819 = 2058316041717
8765543 + 234819 = 9000362
```

4. Lenguaje a utilizar

El lenguaje a utilizar en este trabajo será C con las siguientes extensiones:

- Operadores new y delete.
- Pasaje de parámetros por referencia (uso de &).
- Declaración de tipos como en C++ para registros y enumerados.
- Sobrecarga de funciones.
- Uso de cin y cout.
- Uso del tipo bool predefinido en C++.

5. Qué se espera

Para cada módulo de cabecera (.h) con los prototipos de las operaciones solicitadas, debe entregarse un módulo (.cpp) con la implementación de dichas operaciones. Debe respetarse estrictamente los prototipos especificados, esto es: nombre de la operación, tipo, orden y forma de pasaje de los parámetros y tipo de retorno.

Los **módulos de cabecera pueden** bajarse de la página web del curso. Estos módulos no forman parte de la entrega, y por lo tanto, **no deben ser modificados**. **Los módulos deben funcionar en el ambiente MinGW instalado en facultad**. Se espera que todos los módulos compilen **sin errores** utilizando las flags "-Wall" , "-Werror" y "-O1", se ejecuten **sin colgarse** y den los **resultados correctos**.

6. Forma de la entrega

Se deberá entregar únicamente el siguiente archivo:

- operaciones.cpp

correspondiente a la implementación del sistema especificado. No se podrá entregar otros archivos que no sean estos.

La primera línea del archivo debe contener un comentario (/* ... */) con la cédula del autor, sin puntos ni dígito de verificación. Por ejemplo, si la cédula es 1.234.567-8, la primera línea de cada archivo deberá ser exactamente:

```
/* 1234567 */
```

7. Advertencia sobre el manejo de la memoria

Cuando un programa contiene errores en el manejo de la memoria, su comportamiento puede ser inestable. Esto implica que algunas veces funciona correctamente y otras no. En ciertos casos esto puede inducir a creer (erróneamente) que ciertos programas, que en realidad son incorrectos, funcionan correctamente. Este aspecto es influenciado, entre otras cosas, por el sistema operativo en el que se ejecuten los programas. Recomendamos tener sumo cuidado con este punto y testear los módulos en sistemas operativos Windows XP. CON LA VERSIÓN DE MINGW INSTALADA EN LAS SALAS DE INFORMÁTICA DE LA FACULTAD.

8. Sobre la individualidad del trabajo

El laboratorio es INDIVIDUAL. Los estudiantes pueden estudiar en grupo y consultarse mutuamente, pero NO pueden realizar en grupo las tareas de codificación, escritura, compilación y depuración del programa.

Los trabajos de laboratorio que a juicio de los docentes no sean individuales serán eliminados, con la consiguiente pérdida del curso, para todos los involucrados. Además todos los casos serán enviados a los órganos competentes de la Facultad, lo cual puede acarrear sanciones de otro carácter y gravedad para los estudiantes involucrados.

No existirán instancias en el curso para reclamos frente a la detección por parte de los docentes de trabajos de laboratorio no individuales, independientemente de las causas que pudiesen originar la no individualidad. A modo de ejemplo y sin ser exhaustivos: utilizar código realizado en cursos anteriores u otros cursos, perder el código, olvidarse del código en lugares accesibles a otros estudiantes, prestar el código o dejar que el mismo sea copiado por otros estudiantes, dejar la terminal con el usuario abierto al retirarse, enviarse código por mail, etc. Asimismo, se prohíbe el envío de código al foro del curso, dado que el mismo será considerado como una forma de compartir código y será sancionada de la manera más severa posible.

Es decir que se considera a cada estudiante RESPONSABLE DE SU TRABAJO DE LABORATORIO Y DE QUE EL MISMO SEA INDIVIDUAL. NO CONFIAR en el borrado automático del directorio pub de las salas Windows. Es decir antes de cerrar sesión borrar todos los archivos del directorio.

9. Fecha de entrega

El trabajo debe entregarse el día **lunes 25 de octubre de 2010 antes de las 22:00 horas**. La entrega se realizará mediante un formulario que se habilitará oportunamente en la página web del curso.