# Tarea 3

Algoritmos y lenguaje de programación

1er. Semestre 2009

## Descripción

Esta tarea puede ser realizada en grupos de hasta dos personas. Debe ser entregada el día viernes 12 de junio, antes de las 6pm. Debe entregar un informe impreso detallando el código escrito y las dificultades encontradas en Secretaría de Electrónica antes de las 6pm. Además, suba el código fuente (SÓLO EL CÓDIGO FUENTE!) via InfoAlumno antes de ese día y hora, en la forma de un archivo de formato `rar` ó `zip` con los nombres de los integrantes del grupo (E. g., `Enriquez-Ominami.rar`). No se aceptan tareas atrasadas. La copia en esta tarea implica un 1 para todas las personas involucradas.

Como se mencionó en clases, sus programas serán probados utilizando el compilador Dev-C++, así que es su responsabilidad asegurarse de la ejecución correcta en ese ambiente de trabajo.

## Problemas

1. Escriba un programa en C llamado `tarea03-01.c` que calcule la suma de los 100 números de 50 dígitos contenidos en el archivo `numeros.txt`. Incluya este resultado en su informe escrito.

   Nótese que no puede leer los números directamente vía `scanf()` a un entero porque son mayores que la capacidad máxima de un tipo de dato `int`. Sin embargo, puede usar la línea `scanf("%1d", &dato)` para leer un dígito y almacenarlo en la variable `dato`.

2. Los números hipercomplejos o *cuaterniones* fueron inventados por Sir William Rowan Hamilton (1805-1865), célebre matemático y físico irlandés. Los cuaterniones poseen numerosas e interesantes propiedades matemáticas que han sido aplicadas con éxito a variados problemas de la física y la ingeniería. Recientemente, los cuaterniones han permitido optimizar ciertos métodos de computación gráfica.

   Un cuaternión $q$ se puede escribir de la forma:

   $$q = \alpha + \beta i + \gamma j + \delta k$$

   Para los fines de este problema, represente un cuaternión como la siguiente estructura de datos:

struct cuaternion

```
typedef struct  {
    double alfa;
    double beta;
    double gamma;
    double delta;
} cuaternion;
```

En base a la definición anterior, se le pide implementar las siguientes funciones:

*a*) Implemente una función `int compara(cuaternion a, cuaternion b)` que reciba dos cuaterniones $a$ y $b$ como argumentos, y retorne un valor $1$ si los dos cuaterniones son iguales, y $0$ en caso contrario.

*b*) Implemente las funciones aritméticas `cuaternion suma(cuaternion a, cuaternion b)` y `cuaternion resta(cuaternion a, cuaternion b)` que suman y restan dos cuaterniones, respectivamente.

*c*) Implemente la función aritmética `cuaternion producto(cuaternion a, cuaternion b)`. Para esta última función, considere que $i^2 = j^2 = k^2 = ijk = -1$, que $ij = k = -ji$, $-ik = j = ki$, e $jk = i = -kj$.

*d*) Implemente un función `cuaternion conjugado(cuaternion q)` que reciba como argumento un cuaternión $q$ y retorne su conjugado $q^*$ , definido como $q^* = \alpha - \beta i - \gamma j - \delta k$.

*e*) Implemente una función `double norma(cuaternion q)`, que reciba un cuaternión como argumento y retorne un número de punto flotante con su norma, definida como $\sqrt{qq^*}$, donde $qq^*$ es el *producto interno* del cuaternión.

*f*) Implemente una función `cuaternion inversa(cuaternion q)` de un cuaternión, donde la inversa está definida como: $q^{-1} = \frac{q^*}{qq^*}$. Use esta función para escribir la función `cuaternion division(cuaternion a, cuaternion b)` entre cuaterniones, definida como $a/b = ab^{-1}$.

*g*) Implemente una funcion `void imprime(cuaternion q)` que imprime en pantalla los $4$ componentes de un cuaternion, entre paréntesis y separados por coma, por ejemplo, `(0.453432, 2.00000, 2344.002000, -3.313132)`.

Pruebe sus funciones con el siguiente programa `tarea03-02.c`:

tarea03-02.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

typedef struct  {
    double alfa;
    double beta;
    double gamma;
    double delta;
} cuaternion;

int compara(cuaternion a, cuaternion b);
cuaternion suma(cuaternion a, cuaternion b);
cuaternion resta(cuaternion a, cuaternion b);
```

```
cuaternion producto(cuaternion a, cuaternion b);
cuaternion conjugado(cuaternion q);
double norma(cuaternion q);
cuaternion inversa(cuaternion q);
cuaternion division(cuaternion a, cuaternion b);
void imprime(cuaternion q);

int main(void)
{
    cuaternion x = {0.3, 43.5, -2.21, -0.04};
    cuaternion y = {1.8, -11.90, 3.231, 4.56};

    printf("Los cuaterniones "); imprime(x); printf(" y "); imprime(y);
    if (compara(x, y) == 0) printf(" no"); printf(" son iguales\n");
    printf("La suma de x e y es "); imprime(suma(x, y)); printf("\n");
    printf("La resta de x e y es"); imprime(resta(x, y)); printf("\n");
    printf("El producto de x e y es"); imprime(producto(x, y));
    printf("\n");
    printf("La division x/y es"); imprime(division(x, y)); printf("\n");
    printf("La division y/x es"); imprime(division(y, x)); printf("\n");

    system("pause");

    return 0;
}
```

3. Escriba un programa llamado `tarea03-03.c` que, dado un valor $n$ menor que \$10000, calcule el número total de maneras en que se pueden combinar las monedas de \$1, \$5, \$10, \$50, \$100 y \$500 pesos para sumar $n$.