1. Realizar un programa para imprimir la siguiente secuencia, siendo M y N valores enteros ingresados por el usuario:

```
00    01    02    ...    0N
10    11    12    ...    1N
...
...
M0    M1    M3    ...    MN
```

2. Realizar un programa que recorra todos los números naturales desde 1 hasta 1000 y calcule e imprima la suma de todos los múltiplos de 2, 3, 4, y 5 en distintos acumuladores. (Acum2 acumula la suma de todos los múltiplos de 2, Acum3 acumula la suma de todos los múltiplos de 3, etc.).

3. Pi. Utilizando la siguiente serie numérica propuesta por Leibniz,

$$\frac{\pi}{4} = \sum_1^{\infty} f(n) = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$

   Realice un programa que aproxime Pi.

4. Otra de PI: Aproxime Pi utilizando la siguiente serie numérica:

$$\frac{\pi}{8} = \sum_1^{\infty} f(n) = \frac{1}{1 \bullet 3} + \frac{1}{5 \bullet 7} + \frac{1}{9 \bullet 11} + \ldots$$

$$f(n) = \frac{1}{(4(n-1)+1)(4(n-1)+3)}$$

5. Realizar un programa que sume los primeros N (ingresado por el usuario) números naturales. El programa debe imprimir el resultado de tal suma. (no aplicar n * (n+1) / 2). Encapsule el cálculo de la suma en una función **`int suma(int N);`** (implementar esta función antes de la función **`main`**).

6. La sucesión:

   $X_0 = 1$

   …

   $X_{i+1} = (x_i + a/x_i)/2$

   Converge a la raíz cuadrada de ***a***, si ***a*** es un número real positivo. Escriba un programa que pida al usuario el valor de ***a***, e imprima su raíz cuadrada utilizando la sucesión anterior. (Compare con el resultado de la función **sqrt(a)**). ¿Cómo va a determinar la convergencia? ¿Cuántos pasos necesita el algoritmo para converger? ¿Cómo depende eso del número **a**? Encapsule el cálculo de la sucesión en una función **`double miSqrt(double a);`** (implementar esta función antes de la función **`main`**).

7. A partir del desarrollo en serie de Taylor de e^x y especializándolo en $x=1$ calcular el valor de ***e***. Intentar minimizar la cantidad de operaciones a realizar en cada termino. Encapsule el cálculo de

la serie en una función **`double calculaE();`** (implementar esta función antes de la función **`main`**).

8. Escriba un programa que pida al usuario los valores de ***a***, ***b*** y ***c*** y calcule e imprima las raíces del polinomio $ax^2 + bx + c$ para todos los casos posibles.

9. Diseñe e implemente un programa que solicite al usuario un número (en base 10) y una nueva base (en el intervalo [2,16]), e imprima la representación de ese número usando la base ingresada. Para coeficientes entre 10 y 15 utilice los caracteres entre 'A' y 'F'.

   Ejemplo:

284 en base 16:	11C
284 en base 4:	10130
284 en base 8:	434
284 en base 13:	18B

10. Se desea determinar si un número es primo o no, para lo cual se tendrá en cuenta que un número es primo si no tiene divisores propios mayores a 1 y menores o iguales que su raíz cuadrada. Haga un programa que implemente este método.

11. Suponga la serie:

    $$S = \sum_{1}^{0xFFFFFF} X_1 + X_2 + X_3$$

    donde X_1, X_2 y X_3 son constantes e iguales a:

    $X_1 = 1.126$

    $X_2 = -1.125$

    $X_3 = -0.001$

    Cuanto es el resultado esperado? Haga un programa para verificarlo. Justifique el resultado encontrado.

12. Utilizando la definición del tipo **`Complejo_t`** vista en clase, implemente las siguientes funciones:

```
Complejo_t sumaComplejos(Complejo_t c1, Complejo_t c2);
Complejo_t multiplicaComplejos(Complejo_t c1, Complejo_t c2);
```

13. Dada la siguiente estructura de datos:

```
typedef struct {
    unsigned char red;
    unsigned char green;
    unsigned char blue;
} RGB_t;
```

    a. Implemente la función **`int compositeColor(RGB_t color);`** que retorne el entero que represente el **`color`** dado según la representación del ejercicio 7 de la práctica 3.

    b. Implemente la función inversa **`RGB_t decompositeColor(int compColor);`** que retorne el RGB_t representado por **`compColor`**.