Ejercicios de programación

Nota: En todos los ejercicios se debe utilizar **funciones**.

1. Hacer un programa que lea dos números naturales **desde la línea de comandos** y calcule el máximo común divisor (MCD) y el mínimo común múltiplo (MCM) de ambos. Para el MCD, se debe utilizar una función con la versión *recursiva* del algoritmo de Euclides. Para el MCM se puede utilizar la relación entre MCD y MCM: MCD(a, b) * MCM(a, b) = a*b. Para convertir un string en número entero usar función **atoi** de la librería **cstdlib**

2. Programa que imprima los números primos desde 2 hasta un número especificado por el usuario(algoritmo: criba de eratóstenes)

3. Calcular las raíces (con precisión de 8 decimales) de la ecuación de segundo grado $A\,x^2 + B\,x + C = 0$, para cualquier valor de los coeficientes A, B y C. Se recomienda usar el tipo doublé.
   Probar con los siguientes datos:
   A = 4 B = 3 C = 2
   A = 1 B = 9 C = 4
   A = 1 10^{-2} B = 1 10^{7} C = 1 10^{-2}

4. Escribir un programa que utilice funciones para calcular las siguientes operaciones matriciales:
   - La lectura (uso obligatorio de **arrays dinámicos**) e impresión de una matriz (valores reales, precisión 3 decimales).
   - Cálculo del producto de los valores de la diagonal principal y la suma de los valores de la diagonal secundaria para una matriz A (m x n).
   - Cálculo de las siguientes normas para una matriz A (m x n):

$$\|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^{m} |a_{ij}| \qquad \|A\|_\infty = \max_{1\le i\le m} \sum_{j=1}^{n} |a_{ij}| \qquad \|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2}$$

   - Cálculo del producto matricial de dos matrices (m x p y p x n). Se recuerda que cada elemento $C_{i,j}$ de la matriz resultante se calcula aplicando la siguiente fórmula:

$$C(i,j) = \sum_{k=1}^{p} A(i,k)\cdot B(k,j)$$

   Utilizar como datos los contenidos en el fichero matrand.txt que contiene los valores de:
   número de filas número de columnas
   valores de la matriz por filas

   de dos matrices. Se deberá ejecutar el programa en modo comando redireccionando la entrada y la salida (resmat.txt)

5. Programa que lee un valor para el radio y a continuación un código a seleccionar para calcular
   código = 1 perímetro de la circunferencia (2 π r), código = 2
   superficie del círculo (π r^2),
   código = 3 superficie de la esfera (4 π r^2) y código = 4
   volumen de la esfera (4/3 π r^3).
   El programa repite la lectura hasta que el usuario detenga la ejecución.

6. Programa que determina si un año es bisiesto. (Definición de año bisiesto: Un año es bisiesto si es divisible por 4, excluyendo al que es divisible por 100, pero no si es divisible por 400).

Determinar los años bisiestos en el rango de 1800 – 2200.

7. Programa que calcula el índice de masa corporal (IMC) de una persona e indica su estado:

$$IMC = \frac{peso\ (kg)}{estatura^2(m)}$$

Delgadez moderada	16,00 - 16,99
Delgadez leve	17,00 - 18,49
Normal	18,5 - 24,99
Sobrepeso	25,00 - 27,49
Preobeso	27,50 - 29,99
Obesidad	30,00 - 32,49
Obesidad leve	32,50 - 37,49
Obesidad media	35,5 – 39.99
Obesidad mórbida	≥40,00

El programa repite la lectura hasta que el usuario detenga la ejecución.

8. Dado un entero positivo, calcular su factorial. El programa debe utilizar una función con la **versión iterativa** y otra función con la **versión recursiva** que devuelvan el valor. Indicar ¿cuál es el mayor número que se puede calcular su factorial?.

9. Programa para el cambio de base de un número en decimal a otra base (2 -16) especificada por el usuario. El programa repite la lectura hasta que el usuario detenga la ejecución.

10. Calcular el valor de las siguientes funciones evaluando su expansión en serie para un número determinado de términos:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} \ldots.$$

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

Calcular también el error verdadero, relativo y el error aproximado relativo (ver pags. 14-28 de apuntes para su definición) con respecto al valor de la función respectiva en la librería cmath.

11. Programa que calcula la **interpolación de Lagrange** que es un método para interpolar una serie de N puntos (xi, yi) dados como datos es hacer pasar un polinomio por tales puntos. La solución clásica es usar la fórmula de Lagrange que expresada matemáticamente es:

$$p(x) = \sum_{1 \le j \le N} \left(\prod_{\substack{1 \le i \le N \\ i \ne j}} \frac{x - x_i}{x_j - x_i} \right) y_j$$

Los datos del programa son:
- Número de puntos N, que a su vez es el número de términos del polinomio.
- Puntos a interpolar (x_1, y_1), (x_2, y_2),....,(x_N, y_N).
- Valor x a interpolar.

El resultado es el cálculo de la interpolación según la fórmula anterior.
El programa repite la lectura del valor a interpolar hasta que el usuario detenga la ejecución.
Se pide dos versiones:

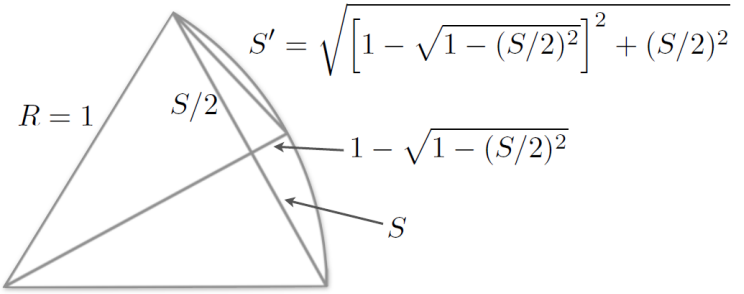
Debe usarse **memoria dinámica** para los arrays.
Uso de la clase **vector**

Los argumentos del módulo son los puntos a interpolar (x_1, y_1), (x_2, y_2),....,(x_N, y_N) y el valor x a interpolar. El resultado es el cálculo de la interpolación según la fórmula anterior.
Aplicarla a 10 puntos equidistantes en el rango [-1,1] obtenidos de la función

$$f(x) = \frac{1}{1 + 25x^2}$$

considerar para la interpolación 500 valores en el rango [-1,1]. Graficar con Excel la función f(x) y los valores obtenidos en la interpolación con Lagrange.

12. Programa que descompone en factores primos un número.
13. Programa que determina si un número es primo
14. Programa que imprime los números primos hasta un número dado (criba de eratóstenes)
15. Métodos de ordenación y búsqueda
16. Cálculo de permutaciones
17. Programa para contar el número de caracteres y blancos, frecuencia de ocurrencia de letras, dígitos
18. Programa para contar la frecuencia de ocurrencia de palabras
19. Suma y producto de polinomios
20. Series de Fourier
21. Cálculo de PI (π) mediante la aproximación de un círculo por polígonos (método de Liu Hui):



$$S' = \sqrt{2 - \sqrt{4 - S^2}}$$

Después de varias iteraciones:

$$\pi \approx \frac{S \cdot N_{sides}}{2}$$

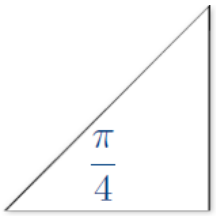
Desarrollar un programa que imprima:

- el número y longitud de los lados (valores iniciales: n= 6, s=1),
- el valor de pi calculado según la formulación anterior y
- el error absoluto entre el valor “verdadero” (math.pi) y el calculado

considerando que en cada iteración se dobla el número de lados hasta un valor dado de iteraciones. Comprobar la evolución del error variando el número de lados (20, 25, 30). Analizar los resultados y explicarlos

22. Cálculo de PI (π) mediante la Fórmula de Leibniz

A partir de la función trigonométrica:



$$\tan\left(\frac{\pi}{4}\right) = 1 \;\rightarrow\; \arctan(1) = \frac{\pi}{4}$$

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \ldots$$

$$\rightarrow\; 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots \;=\; \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} \;=\; \frac{\pi}{4}$$

A partir de la función trigonométrica:

Desarrollar un programa que para 1000, 100000, 1000000 términos:

- calcule pi según la serie,
- imprima cada 100 términos el valor del término, pi calculado según la formulación anterior y
- el error absoluto entre el valor “verdadero” (math.pi) y el calculado

23. Otra operación común en el tratamiento de imágenes es la obtención del **histograma** de intensidades de los pixels, consistente en hallar, por cada valor posible de la intensidad (0 - 255), el número de pixels que tienen tal valor (frecuencia).Se pide escribir un programa que lea imagen (matriz NxM de pixels) e imprima el histograma de la imagen de los pixels con frecuencia diferente de cero.

24. Desarrollar un programa para la gestión de una lista. El contenido de los nodos de la lista puede ser un entero. Como mínimo debe tener funciones para añadir un nodo al final de la lista y mostrar el contenido de la lista.

25. Calcular la derivada de la función:

$$f(x) = x^2 + \exp(x) + \log(x) + \sin(x)$$

para el valor 0.5, usando la fórmula: $f'(x) \approx \frac{f(x+h) - f(x)}{h}$ para $h \rightarrow 0$

utilizar como valor inicial de h = 1E-2 y dividirlo por 10 en cada iteración mientras h > 1E-15. Imprimir el valor exacto y para cada iteración: el valor de h, el valor numérico y el valor absoluto de la diferencia. Es fácil comprobar que el valor exacto de la derivada es:

$$f'(x) = 2x + \exp(x) + \frac{1}{x} + \cos(x)$$

Analizar los resultados.

Nota:
Para un valor pequeño h, la expansión de Taylor es:

$$f(x+h) \approx f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + \ldots$$

A partir de la cual se obtiene la formula de la derivada aproximada:

$$\frac{f(x+h)-f(x)}{h} \approx f'(x) + \frac{h}{2}f''(x) + \frac{h^2}{6}f'''(x) + \ldots$$

Es decir, se tiene un error de truncamiento (aproximación) de O(h), pero hay un error de redondeo relacionado con la precisión de la máquina:

$$f(x+h) \approx f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + \ldots + \epsilon_m$$

Si se tiene en cuenta en la derivación numérica:

$$f'_{\text{numerical}}(x) = \frac{f(x+h)-f(x)}{h} \approx f'(x) + \left[\frac{h}{2}f''(x) + \frac{h^2}{6}f'''(x) + \ldots\right] + O\left(\frac{\epsilon_m}{h}\right)$$

El error total es: $\sim O(h) + O\left(\frac{\epsilon_m}{h}\right)$

Para un número en doble precisión: $\epsilon_m \approx O(10^{-15}) - O(10^{-16})$

Por lo que el valor de h para que el error total se minimice: $h \approx O(\sqrt{\epsilon_m}) \approx O(10^{-8})$

26. Programa que imprime el día de la semana para una fecha (dia, mes, año) dada, que se **ingresa desde la línea de comando**. Las fórmulas para determinar el día de la semana son:
$y_0 = y - (14 - m) / 12$
$x = y_0 + y_0/4 - y_0/100 + y_0/400$
$m_0 = m + 12 * ((14 - m) / 12) - 2$
$d_0 = (d + x + (31*m_0) / 12) \bmod 7$
El resultado d_0 indica 0 para Domingo, 1 Lunes, …

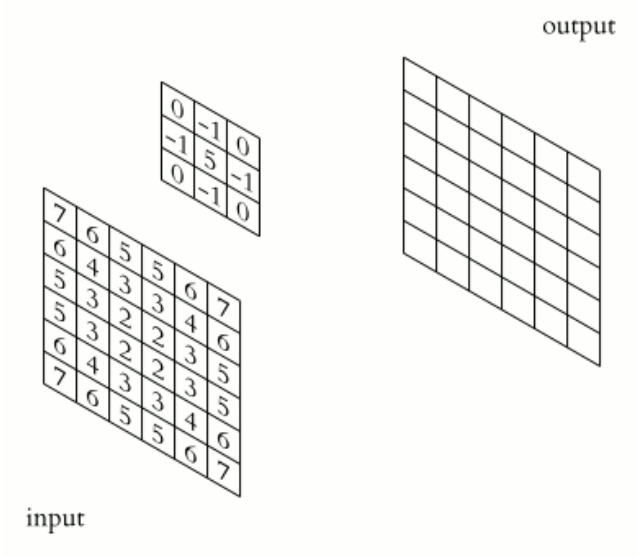
27. Programa y resultados de ejecución para obtener la tabla de verdad de los siguientes circuitos lógicos combinacionales:
ABCD + A!B!C + B!C!D
AB + CD + ABC + BCD

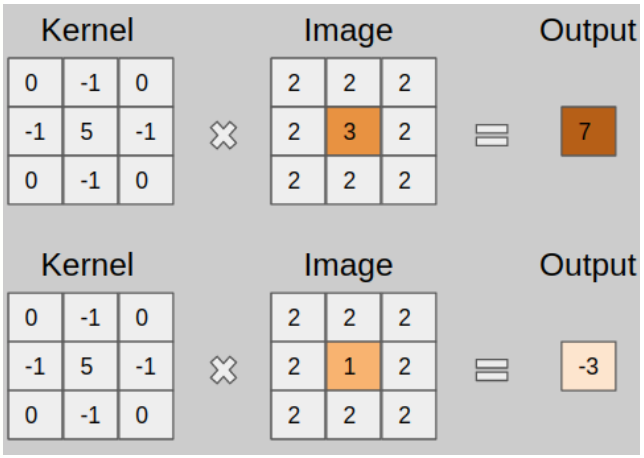
28. Una **imagen** (tipo mapa de bits) se considera como una matriz de NxM pixels.Un **pixel** tiene asignado un nivel de gris, que es un número entero sin signo en el rango de 0 a 255 (escala de grises).

La operación de convolución para una imagen I y un kernel *h* se obtiene según la siguiente fórmula:

$$J(r,c) = [I * h](r,c) = \sum_{\rho=-\infty}^{\infty}\sum_{k=-\infty}^{\infty} I(r-\rho, c-k)h(\rho,k)$$

Un ejemplo de aplicación es:





Para la operación de ***detección de bordes*** se asigna el valor de intensidad de cada pixel interior y se realiza una convolución aplicando el siguiente kernel:

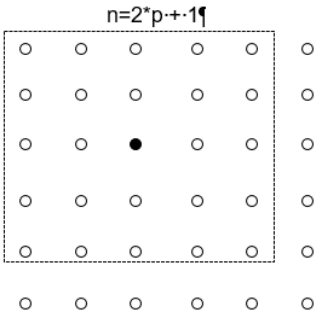
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Para la operación de ***resaltado*** se asigna el valor de intensidad de cada pixel interior y se realiza una convolución aplicando el siguiente kernel:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

La operación para eliminar el ruido en una imagen es la de ***suavizado***, que consiste en asignar el valor de intensidad de cada pixel como el valor medio de la intensidad de los pixels en un entorno del pixel (**p**) incluido él mismo (ver figura)
Para entornos cuadrados el promedio resulta de aplicar:

$$f'(x,y) = \frac{1}{n^2} \sum_{i=-p}^{p} \sum_{j=-p}^{p} f(x+i, y+j)$$



donde: n = 2*p + 1 (entorno de nxn pixels) y p = 1, 2, 3, …
En los bordes hay que asegurar que los índices de los pixels de la imagen estén en el rango correcto y dividir sólo por el número de pixels que cumplen tal condición.

Se pide escribir un programa que:

- Lea una imagen en formato .pgm consistente en:
  - Dos caracteres seguidos
  - Dos números enteros que representan el ancho y alto de la imagen
  - Un número con el valor de gris máximo
  - Los valores de gris de los pixels en formato matricial y con especificación de formato %-5d

- Imprima en formato .pgm la imagen (matriz) a la que se ha aplicado la detección de bordes, el resaltado y el suavizado (usar p = 1). Los resultados se deben redireccionar a los ficheiros resdetbo.txt, resresa.txt, ressuav.txt respectivamente.
- Imprima, mediante el uso de una función, el ***histograma*** de intensidades de los pixels. Un histograma contiene para cada valor de la intensidad (0 - 255), el número de pixels que tienen tal valor (frecuencia). El resultado se debe redireccionar al fichero reshisto.txt

Usar para las pruebas los siguientes datos de prueba. Para la aplicación real usar la siguiente imagen. Para visualizarlo utilizar el programa nomacs que permite leer ficheros em formato pgm.

29. El método de **Newton-Raphson** se utiliza para aproximar una raíz real de una función. Requiere de un valor inicial de la raíz x_0 y genera aproximaciones a la raíz

$$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$$

donde $f'(x_j)$ es la derivada de la función *f* evaluada en el punto $x = x_j$. Las iteraciones terminan si se cumple alguna de las siguientes condiciones:

- El criterio de convergencia del método se alcanza: $\left|x_j - x_{j-1}\right| < \varepsilon$
- Se supera un número máximo de iteraciones.
- El valor de la derivada es cero.

Se pide escribir un módulo seguro para implementar el método de Newton-Raphson descrito antes, contemplando el paso de funciones como argumentos de funciones y la devolución del número de iteraciones.
Para probar el módulo, calcular la raíz de

$$f(x) = x^4 - 6.4x^3 + 6.45x^2 + 20.538x - 31.752$$

usando los valores iniciales [1, 2, 3, 3.5, 4.2]. Analizar los resultados.

30. Desarrollar un módulo que implemente el método de Runge-Kutta de cuarto orden para resolver una ecuación diferencial de primer orden y' = F(x,y) con valor inicial y(a) = α. Operaciones:

$$\mathbf{K}_0 = h\mathbf{F}(x, \mathbf{y})$$

$$\mathbf{K}_1 = h\mathbf{F}\left(x + \frac{h}{2}, \mathbf{y} + \frac{\mathbf{K}_0}{2}\right)$$

$$\mathbf{K}_2 = h\mathbf{F}\left(x + \frac{h}{2}, y + \frac{\mathbf{K}_1}{2}\right)$$

$$\mathbf{K}_3 = h\mathbf{F}(x + h, \mathbf{y} + \mathbf{K}_2)$$

$$\mathbf{y}(x + h) = \mathbf{y}(x) + \frac{1}{6}(\mathbf{K}_0 + 2\mathbf{K}_1 + 2\mathbf{K}_2 + \mathbf{K}_3)$$

Los parámetros que se especifican son la función con la ecuación diferencial, el valor de inicio de la integración, el valor inicial de y, el valor final de x y el valor h de integración.
Aplicar el módulo para resolver:

$$\frac{dy(t)}{dt} + 2y(t) = 3e^{-4t}$$

$$y(0) = 1$$

$$t_0 = 0, \quad t_{\max} = 4$$

considerando los siguientes pasos h: [0.1, 0.01, 0.001]. Graficar con Excel los resultados correspondientes a los valores calculado y exacto: $y(t) = \frac{5e^{-2t} - 3e^{-4t}}{2}$