INFORMÁTICA

Curso 16/17 **Enunciados de ejercicios** Grupos Mec./Diseño

Capítulo - 2. Primeros programas

1. Identifica y corrige los errores si existen, en los siguientes apartados

a) `printf("El valor de la variable es %d \\n", &entero)`

b) `scanf("%i %i" &valor1 valor2);`

c) Con el siguiente trozo de programa, la salida que se obtiene es "`El valor de entero es 12`"

```
entero = 12;
if(entero = 7){
printf("El valor de entero es 7);
} else {
printf("El valor de entero es 12);
}
```

d) Si se sustituye el *bloque* **`while`**

```
cont = 1;
while (cont <= n){
cont = cont + 1;
suma = suma + cont;
}
```

por el existente en el del código **??**, se obtiene el mismo resultado.

2. Dada la sentencia

```
resultado = a * b * c;
```

Escribe un programa que pida al usuario tres números enteros, utilice la sentencia anterior y escriba por pantalla el resultado.

3. Escribe un programa que pida un número entero y determine si es par o impar. Una ejecución típica debería ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número entero -> -243
El número -243 es IMPAR
```

Ahora haz otro programa que determine si el número entero es múltiplo de tres. Si no lo es, que determine su resto. Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número entero -> 25
El número 25 NO es un múltiplo de 3. Su resto es 1.
```

4. Siguiendo los problemas anteriores, ahora se pide que escribas un programa que pida un número entero y a continuación otro entero. Que compruebe si éste último es un divisor del primero. En caso contrario que determine el resto, de la división entera. Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número entero -> 25
Introduce un segundo entero (posible divisor del anterior) -> 7

El número 25 NO es un múltiplo de 7. Su resto es 4.
```

5. Escribe un programa que intercambie los respectivos valores que contengan dos variables. Esto es, si la variable $a = -12$ y la variable $b = 7$, entonces el código tiene que ser capaz de obtener la salida: $a = 7$ y $b = -12$. Una posible ejecución podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un valor para la primera variable a = 27
Introduce un valor para la segunda variable b = -53

Después del intercambio:

El valor de la variable a = -53 y el de la variable b = 27
```

6. Describe qué hace el siguiente *bloque* **`while`**:

```
var1 = 1;
var3 = 1;
var2 = 25;
while (var1 < var2){
var1 = var1 + 1;
var3 = var3 * var1;
}
```

7. Escribe un programa que calcule el factorial de número natural, dado. Una posible ejecución sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
```

```
**************************

Introduce el número natural -> 7

El factorial de 7 es 7! = 5040
```

8. Escribe un programa que calcule la potencia entera positiva de un número entero. Esto es, si $a \in \mathbb{Z}$ es la base y $b \in \mathbb{N}$ el exponente, se pide a^b. Una posible ejecución sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número entero (base de la potencia) -> -2
Introduce ahora el exponente (número natural) -> 5

La potencia 5 de -2 es -32
```

9. Escribe un programa que sume los dígitos de un número natural. Esto es, dado el número 7435 la suma de sus dígitos es 19. La idea es ir obteniendo el resto de la división entera por 10. Por ejemplo, siguiendo con el número anterior: $7435 \,\%\, 10 = 5$ y $7435/10 = 743$. Seguidamente se hace el mismo proceso pero ahora con 743. Es decir: $743 \,\%\, 10 = 3$, que será otro dígito que se suma al anterior y se opera: $743/10 = 74$. A continuación se sigue una pauta análoga con 74 y así sucesivamente, hasta que la parte entera sea cero. Una posible ejecución del programa sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número natural -> 45207

La suma de los dígitos de 45207 es 18
```

10. A partir del programa anterior, escribe un código que escriba el reverso de un número natural. Esto es, si se introduce el número 45178, la salida debe ser 87154. A continuación que compruebe si es *capicúa* o no. Una posible ejecución del programa sería

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número natural -> 45207

El reverso de 45207 es 70254

En número 45207 NO es capicúa
```

11. Escribe un programa que determine si un número es *perfecto*. Se dice que un número natural es *perfecto* si su valor coincide con la suma de sus divisores. Por ejemplo, el número 6 es *perfecto* ya que $6 = 1 + 2 + 3$. También lo son 28 y 496. La idea del algoritmo es muy sencilla: basta con tomar todos los números menores

que el introducido y comprobar si alguno es divisor. Si lo es, entonces se suma con los anteriores que han sido divisores. Al final la suma total y el número natural inicial deben ser iguales, si el número es *perfecto*.

Utilizando el código anterior, haz un programa que dado un número natural, saque por pantalla todos los *números perfectos* menores que él.

12. Haz un programa que calcule el *Máximo común divisor (m.c.d.)* y el *Mínimo común múltiplo (m.c.m.)* de dos números naturales.

NOTA:

Para calcular el *máximo común divisor (m.c.d.)* de dos números enteros positivos, puede aplicarse el *Algoritmo de Euclides*. Este proceso se basa en que dados dos números D (*dividendo*) y d (*divisor*) con $D > d$, la división entre ambos números se expresa de la forma:

$$D = C \cdot d + r$$

donde C es el *cociente* y r el *resto*. Si k es un divisor de D y d, entonces estos valores serán de la forma: $D = T \cdot k$ y $d = s \cdot k$ y como consecuencia, k también debe ser un divisor de r, ya que:

$$T \cdot k = C \cdot s \cdot k + r \iff k \cdot (T - C \cdot s) = r$$

Teniendo en cuenta este hecho, se hace a continuación, la división entre d y r, pero ahora d como *dividendo* y r como *divisor*, obteniéndose:

$$d = C_2 \cdot r + r_2$$

Si $r_2 \neq 0$, entonces de nuevo, si k es divisor de d y r, también lo será de r_2. Con lo cual se vuelve a repetir el mismo procedimiento, dividiendo r entre r_2 y así sucesivamente hasta que el resto de las sucesivas divisiones sea cero, para obtener el *m.c.d.*

Basándonos en este hecho se plantea un proceso iterativo, llamado el *Algoritmo de Euclides*, cuyo desarrollo se presenta en el siguiente seudocódigo:

Seudocódigo: Algoritmo de Euclides

```
Dados a y b (a > b)
Calcula el m.c.d.(a,b)

D <- a; d <- b;
Mientras r != 0 haz lo siguiente
r <- mod(D,d)  //resto de la división entera
D <- d
d <- r
fin
el resultado es: D
```

Como ejemplo y antes de resolver el problema aplica este algoritmo, para determinar el *m.c.d.*(35, 25).

Capítulo-3. Introducción al lenguaje C

13. Realiza un programa que muestre, para cada *tipo* definido en el capítulo, el número de *bytes* que tiene asignado, junto con el rango y la precisión. Por ejemplo, parte de la salida podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************
```

```
..........
...........
..........
El tipo float ocupa 4 bytes
El rango de sus valores es de 1.17549e-038 a 3.40282e+038
Su precisión es de 6 cifras significativas
.....................
...................
```

14. Escribe un programa que dado un carácter, obtenga su correspondiente código ASCII en decimal, octal y hexadecimal. Un ejemplo de ejecución del código podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un carácter -> c
El carácter c tiene el código ASCII: 99 decimal, 143 octal, 63 hexadecimal
```

Seguidamente, haz otro programa que ejecute el proceso inverso esto es, que se pueda introducir un valor entre 0 y 255 y obtenga el correspondiente carácter. Analiza si existe alguna diferencia entre utilizar **`char`** y **`unsigned char`**.

15. Realiza un programa que determine que para **C**, cualquier valor distinto de 0 es *verdadero*, pero si es 0, entonces es *falso*.

16. La relación que transforma los *grados centígrados* a *grados Fahrenheit* es

$$F = \frac{9}{5}\,C + 32$$

Se pide que hagas un programa que pida *grados centígrados* y los transforme en *grados Fahrenheit*. Haz otro programa que realice el proceso inverso.

17. El *Índice de Masa Corporal (IMC) o Body Mass Index*:

$$\text{IMC} = \frac{\text{Peso en Kg.}}{\text{Altura en metros}^2}$$

da una estimación sobre el peso correcto según la altura. Se pide que realices un programa en que dados un peso y una altura, se calcule el IMC. Una posible ejecución podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce tu peso en Kg. -> 70.2
Introduce tu altura en Mts. -> 1.73

Tu Índice de Masa Corporal es 23.46
```

18. Escribe un programa en el que se introduzcan las coordenadas cartesianas de un punto del plano y devuelva por pantalla su distancia al origen.
(**Nota:** se pueden utilizar las funciones matemáticas del **C**).

19. Escribe un programa en el que se introduzcan las coordenadas de un punto y devuelva sus coordenadas polares.
(**Nota:** se pueden utilizar las funciones matemáticas del **C**).

20. Dada la siguiente instrucción

```
while(( sal = scanf("%i %i", &ent1, &ent2 ) ) != 2){
..........
BUCLE
..........
}
```

a) Indica qué es lo que hace.

b) ¿Cuándo se ejecuta el bucle?.

21. Haz un programa que lea un entero *sin signo* y lo imprima, pero debe comprobar que la entrada es correcta. Esto es, si se introduce un carácter, ha de tenerse la posibilidad de repetir la entrada, sin detener la ejecución del programa. Un posible ejemplo sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un entero sin signo-> x

Disculpa, debes introducir un ENTERO SIN SIGNO

Introduce un entero sin signo-> 134

El entero sin signo introducido es 134
```

¿qué sucede si introduces un entero *con signo*? ¿y un valor con decimales?

22. Siguiendo con la idea del problema anterior, haz un programa que lea dos números decimales y los imprima (por supuesto, comprobando que la entrada es correcta). A continuación que el programa lea dos caracteres (con comprobación) y que los imprima. Por último, que lea un entero (también con comprobación) y que lo imprima.

23. Utilizando las funciones **`getchar(), putchar()`** y la instrucción **`while`** ¿serías capaz de hacer un programa que leyese un texto del teclado y lo imprimiese en la pantalla? ¿y además que contase el número de caracteres introducido?.

24. Sabiendo el tamaño de bytes de memoria que reserva un tipo **`unsigned long long`**, haz un programa que realice una tabla con los factoriales del 1 al 25 y se almacene una una variable de este tipo. El resultado debe volcarse en un fichero llamado `Datos.dat`. ¿Qué observas del resultado?. Repite el mismo programa pero almacenando los factoriales en una variable decimal **`double`**. Compara los resultados, ¿puedes sacar alguna conclusión?

Capítulo-4. Control de flujo

25. Anteriormente ya se ha definido el *Índice de Masa Corporal*. Se pide que hagas un programa que una vez calculado este coeficiente, tenga una salida que se ajuste al cuadro 1 utilizando **`if...else`** múltiple. Una posible ejecución podría ser

Resultados del Índice de Masa Corporal	
Peso bajo	IMC $< 18{,}5$
Peso normal	$18{,}5 \leqslant$ IMC < 25
Sobrepeso	$25 \leqslant$ IMC < 30
Obesidad	$30 \leqslant$ IMC

Cuadro 1: Interpretación del Índice de Masa Corporal

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce tu peso en Kg. -> 70.2
Introduce tu altura en Mts. -> 1.73

Tu Índice de Masa Corporal es 23.46 y tu peso es ADECUADO.
```

26. Realiza un programa de forma que introducido un año, determine si es bisiesto o no. Para resolver este ejercicio se puede consultar la página *Leap Year* de Wikipedia en inglés.

27. Realiza un programa que escriba en un fichero, una tabla con las potencias cuadradas y cúbicas de los n primeros números naturales, donde n debe ser seleccionado desde el teclado (ha de cumplirse que $n \leqslant 100$). Este resultado debe escribirse en un fichero llamado `Datos.dat`. Una posible ejecución del programa podría ser

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce el tamaño de la tabla -> -15
Has de introducir un número positivo menor o igual a 100
Introduce de nuevo, el tamaño de la tabla -> 5
```

y una posible solución sería

```
N^1         N^2         N^3
1           1           1
2           4           8
3           9          27
4          16          64
5          25         125
```

28. **Razón áurea.** Calcula el valor de la *fracción continua*

$$1+\cfrac{1}{1+\cfrac{1}{\ddots\cfrac{}{\ddots\cfrac{}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+1}}}}}}}}$$

La entrada del programa debe ser un valor entero que determine la cantidad de términos que deben calcularse.

Se trata de una sucesión que se aproxima al ***número áureo*** o ***razón áurea***: $(1+\sqrt{5})/2$. A continuación se pide que hagas un programa que dado un valor (por ejemplo: 10^{-6}), se determine el número de términos de la sucesión que se necesitan, para obtener una aproximación de la *razón áurea*, menor que el valor introducido. En la salida también debe darse esta aproximación.

29. Escribe un programa que determina todos los divisores de un numero natural y el resultado lo escriba en un fichero llamado `Datos.dat`. Una posible ejecución podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce el número natural -> -72
El número debe ser positivo
Introduce de nuevo, el número natural -> 64
```

y la salida en el fichero `Datos.dat` debe ser

```
Los divisores de 64 son:  1 2 4 8 16 32 64
```

30. Con las funciones **`getc`** y **`putc`** escribe un programa, que lea un texto de un fichero de entrada (por ejemplo `Entrada.dat`) y escriba dicho texto, en un fichero de salida (por ejemplo `Salida.dat`). Se debe poder contar el número de caracteres del texto (incluidos los *blancos*). Por ejemplo, el fichero `Entrada.dat` podría contener el texto:

```
Esto es un texto para probar
que el programa funciona
```

31. Utilizando un bucle **`for`**, escribe un programa que calcule el factorial de un número natural n, introducido por el teclado. Ten en cuenta, que el valor del factorial para números mayores de 21 no es muy precisa[1], debido a un posible desbordamiento (*overflow*). Una posible ejecución del programa podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
```

[1] Condicionado por las especificaciones propias del compilador DEV-C++ utilizado.

```
**************************

Introduce el número natural -> 59
Introduce un Número Natural menor o igual que 21!!!
Introduce de nuevo, el número natural -> 15
```

y el resultado

```
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 = 15 ! = 1307674368000
```

32. **Variaciones sin repetición**. Se llama *variaciones de m elementos tomados de n en n* ($0 \leqslant n \leqslant m$) al número

$$V_{m,n} = m \cdot (m-1) \cdots (m-n+1) = \frac{m!}{(m-n)!}$$

Haz un programa consistente en el cálculo de las variaciones, en el que los valores m y n, deben ser introducidos por teclado y pueden ser introducidos en cualquier orden.

33. **Variaciones con repetición**. Se llaman *variaciones con repetición de m elementos tomados de n en n* al número

$$VR_{m,n} = m^n$$

Haz un programa, empleando la instrucción **`for`**, pero sin utilizar la función **`pow`** (de la biblioteca de funciones matemáticas), que calcule las variaciones con repetición de m elementos tomados de n en n. Los números m y n serán introducidos por teclado y en cualquier orden.

34. **Combinaciones**. Se llaman *combinaciones de m elementos tomados de n en n* ($0 \leqslant n \leqslant m$) al número

$$\binom{m}{n} = \frac{V_{m,n}}{n!} = \frac{m!}{(m-n)!\,n!}$$

Haz un programa que calcule el número combinatorio correctamente, dados dos números naturales m y n que entren por teclado y en cualquier orden[2].

35. Mediante la instrucción **`do--while`**, haz un programa que compruebe si un número introducido por teclado es un entero sin signo. Si lo es, que lo imprima. En caso contrario que vuelva a pedirlo por teclado.

36. Anteriormente se planteó el *Algoritmo de Euclides*. Ahora se pide que programes este algoritmo con la instrucción **`do--while`**.

37. Escribe un programa que determine si un número entero positivo n es primo o no. Por lo general, basta con comprobar si todos los números menores o iguales que $\sqrt{n}$, dividen a n.

38. Utilizando una estructura **`switch`** escriba un programa que dados dos números enteros positivos, permita elegir al usuario, si obtener $V_{m,n}$, $VR_{m,n}$ o $\binom{m}{n}$. El programa terminará cuando se introduzca el carácter **`@`**.

39. Realiza un programa que imprima un cuadrado de asteriscos en un fichero llamado `Datos.dat`. El dato de entrada n, será el tamaño del lado. Por tanto el cuadrado contendrá $n \times n$ asteriscos. Un ejemplo de ejecución sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce el tamaño del lado del cuadrado -> -25
Debe ser un entero positivo
Introduce, de nuevo, el tamaño del lado (entero positivo) -> 5
```

[2]Este programa se puede simplificar definiendo el concepto de función que se verá más adelante

con una salida de la forma

```
*****
*****
*****
*****
*****
```

a) Con la misma idea que en el caso anterior, ahora que imprima, para un tamaño n, un triángulo de forma:

```
&
&&
&&&
&&&&
&&&&&
&&&&&&
```

(en este caso $n = 6$)

b) Ahora un triángulo de la forma

```
$$$$$$
$$$$$
$$$$
$$$
$$
$
```

($n = 6$)

c) El siguiente con la forma

```
######
#####
####
###
##
#
```

($n = 6$)

d) Por último, una pirámide de la forma:

```
@
@@@
@@@@@
@@@@@@@
@@@@@@@@@
@@@@@@@@@@@
```

($n = 6$)

40. Se llama *triángulo de Floyd* al triángulo rectángulo formado con números naturales. Para crear un *triángulo de Floyd*, se comienza con un 1 en la esquina superior izquierda, y se continúa escribiendo la secuencia de los números naturales de manera que cada línea contenga un número más que la anterior. Por ejemplo, el *triángulo de Floyd* para $n = 4$ sería:

1

2 3

4 5 6

7 8 9 10

Se pide que hagas un programa que dado un número n (tamaño del cateto), se imprima el *triángulo de Floyd*, en un fichero.

41. Realiza un programa que implemente el *método congruencial lineal* y obtenga 20 números aleatorios. La salida debe realizarse a un fichero llamado **`Salida.dat`** y se debe poder guardar los resultados obtenidos con ejecuciones anteriores. Prueba con los valores $a = 5, b = 3, m = 16$ y $n_0 = 7$. Repite la ejecución cambiando los valores de la semilla.

42. Realiza un programa que determine los n números enteros *pseudo-aleatorios* en el intervalo $[M, N]$ cuyos extremos deben introducirse por teclado y la salida debe escribirse en un fichero. Se debe introducir también el valor de la semilla y observar en el fichero las diferentes sucesiones obtenidas para diferentes semillas. Implementar la obtención de la semilla mediante la lectura del reloj del sistema.

43. Realiza un programa que determine los n números reales *pseudo-aleatorios* en el intervalo $[a, b]$ cuyos extremos deben introducirse por teclado y la salida debe escribirse en un fichero. Se debe introducir también el valor de la semilla y observar en el fichero las diferentes sucesiones obtenidas para diferentes semillas. Implementar la obtención de la semilla mediante la lectura del reloj del sistema.

44. Realiza un programa que simule el lanzamiento de una moneda $1{,}000{,}000$ de veces y obtener la frecuencia absoluta para cada una de las distintas caras.

45. Realiza un programa que simule el lanzamiento de un dado $6{,}000{,}000$ de veces y obtener la frecuencia absoluta para cada una de las distintas caras. ¿Cuál sería la probabilidad de cada una de las caras?

46. Utiliza uno de los dos algoritmos de Box-Muller planteados en teoría, para simular un conjunto de números aleatorios con distribución Normal $N(\mu, \sigma)$, cuyos valores μ y σ deben introducirse por teclado así como, el número de datos que se deben generar.

47. Comprueba, tal y como dice en la teoría, que el segundo algoritmo de Box-Muller es más eficiente que el primero.

48. Anteriormente has realizado un programa que calculaba la tabla de los cuadrados y cubos de los n primeros números naturales. Amplía este programa para que también te pida la potencia. Por ejemplo, una posible ejecución podría ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce el número de potencia máxima -> 5

Introduce la longitud de la tabla -> 10
```

y la salida aproximadamente:

```
N^1        N^2        N^3         N^4         N^5
1          1          1           1           1
2          4          8          16          32
```

```
3          9         27         81        243
4         16         64        256       1024
5         25        125        625       3125
6         36        216       1296       7776
7         49        343       2401      16807
8         64        512       4096      32768
9         81        729       6561      59049
10       100       1000      10000     100000
```

(Propuesto en examen: julio – 2015)

49. Se pide que a partir de un intervalo dado se generen una cierta cantidad de números aleatorios ***enteros***: $x_1, \ldots, x_n$ y con ellos calcules la *media* y la *cuasi-desviación típica muestral*, esto es:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{y} \qquad s = \sqrt{s^2}$$

donde

$$s^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - n\overline{x}^2\right)$$

El programa debe permitir al usuario seleccionar:

a) la cantidad de números *pseudo-aleatorios* que deben generarse

b) los extremos del intervalo (que deben ser valores ***enteros***), en cualquier orden.

c) semilla (un valor entero), para generar diversas muestras de números *pseudo–aleatorios*.

Una ejecución típica del programa con salida en pantalla sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce la cantidad de números a generar ( > 0) -> -7
Introduce la cantidad de números a generar ( > 0) -> 100

Introduce los extremos del intervalo (enteros), en cualquier orden -> 10 1


Introduce la semilla -> -3

******** Salida de resultados ********

El intervalo es [1, 10]
La semilla es -3
El número de datos es 100:

La media es 5.690000000
La CuasiDesviación Típica muestral es 2.751289872
```

(Propuesto en examen: curso 15/16)

50. Dado un número ***real*** x definido de la forma

$$x = 1 + \left(\frac{1}{2}\right)^n \qquad \text{con} \quad n \in \mathbb{N}$$

se pide que realices un programa que determine el mayor número natural n_0 tal que se cumpla:

$$x = 1 + \left(\frac{1}{2}\right)^n > 1$$

El programa debe permitir al usuario seleccionar si el cálculo se realiza en *simple precisión*, tecleando la letra **`S`** o en *doble precisión*, tecleando la letra **`D`**.

Una ejecución típica del programa con salida en pantalla sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce S (simple precisión) o D (doble precisión) -> P
Introduce S (simple precisión) o D (doble precisión) -> S

El valor de n0 es: 95
El valor de 1/2 elevado a n0 es: 0.0023430127
```

(Propuesto en examen: curso 15/16)

51. Programa un código en el que la computadora genere un número entero aleatorio entre 1 y 100 y el usuario lo adivine. Cada intento del usuario introduciendo un número por teclado debe tener respuesta orientativa de la computadora (el número introducido es mayor/menor).

- El número debe ser generado con semilla variable en base al reloj del sistema.
- El usuario debe tener la opción de abandonar a mitad del juego.
- Cuando el usuario adivine el número se le informará de los intentos que ha realizado.
- Cuando haya ganado se ofrecerá al usuario la posibilidad de volver a jugar (nueva ejecución), de manera que tras varias partidas se le informe de la cantidad media de intentos que ha hecho a lo largo de todas las ejecuciones.
- El número de intentos de cada juego, el número de ejecuciones y la media de intentos se grabarán en un fichero cuyo nombre será introducido por teclado.

Un ejemplo de la salida en pantalla podría ser:

Apellidos:

Nombre:

Nº de Matrícula:

****

Introduce el número natural que crees que es (0 para terminar): 20

El número introducido es menor.

Introduce el número natural que crees que es (0 para terminar):30

El número introducido es mayor.

Introduce el número natural que crees que es (0 para terminar):25

Has adivinado en 3 intentos. ¿Jugamos de nuevo?

(Propuesto en examen: curso 15/16)

52. Programa un código que implemente el algoritmo para calcular la potencia enésima (con $n \in \mathbb{N}$) de la suma de dos números enteros mediante la expresión

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}.$$

- Los datos (a, b y n) deben introducirse por teclado.
- Los datos y los resultados se grabarán en un fichero cuyo nombre será introducido por teclado.
- Se presentarán por pantalla los distintos factores de cada sumando.

Un ejemplo de la salida en pantalla podría ser:

Apellidos:

Nombre:

Nº de Matrícula:

****

Introduce dos números enteros: -5 12

Introduce la potencia: -2

Introduce la potencia: $^{(1)}$ 2

Las potencias de $a = -5$ son: - - - -

Las potencias de $b = 12$ son: - - - -

Los números combinatorios son: - - - -

El resultado es: - - - -

$^{(1)}$ Control de entrada de dato correcto.

(Propuesto en examen: curso 15/16)

Capítulo-5. Datos estructurados

53. Dado un vector real de doble precisión:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$$

se pide, que realices un programa que permita:

a) Calcular el valor de las normas:

$$\|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x} \cdot \boldsymbol{x}}; \qquad \|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|; \qquad \|\boldsymbol{x}\|_\infty = \max_{1 \leqslant i \leqslant n} |x_i|$$

b) Invertir las componentes del vector inicial.

c) Ordenar de menor a mayor, las componentes del vector inicial.

Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
*************************

Introduce la dimensión del vector -> -1
Por favor introduce un número positivo !!!! -> 5
Introduce V[1] = 0
Introduce V[2] = -1
Introduce V[3] = 1
Introduce V[4] = 3
Introduce V[5] = -6
```

La salida típica en un fichero de datos debe ser:

```
El vector introducido  es:
V[1] = 0.00; V[2] = -1.00; V[3] = 1.00; V[4] = 3.00; V[5] = -6.00;
La norma 2 es: 6.85565

La norma 1 es: 11.00000

La norma infinito es: 6.00000

El vector invertido es:
V[1] = -6.00; V[2] = 3.00; V[3] = 1.00; V[4] = -1.00; V[5] = 0.00;

El vector ordenado es:
V[1] = -6.00; V[2] = -1.00; V[3] = 0.00; V[4] = 1.00; V[5] = 3.00;
```

54. Escribe un programa que abra un fichero con un nombre especificado por el usuario y añada lo que el usuario teclea.

55. Dada una cadena de caracteres escribe un programa que cuente el número de caracteres e invierta la frase (debe aparecer escrita al revés). Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
*************************

Introduce el nombre del fichero de salida -> Datos.dat

El nombre del fichero introducido es: Datos.dat

Introduce la frase -> El valle estaba bonito y florido
```

La salida típica en un fichero de datos debe ser:

```
La frase introducida es ' El valle estaba bonito y florido '

La frase contiene 32 caracteres.

La frase invertida es ' odirolf y otinob abatse ellav lE '.
```

(Propuesto en examen: curso 2014/15).

56. ***Producto de dos polinomios.*** Sean los polinomios

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$
$$q(x) = b_m x^m + b_{m-1} x^{m-1} + \cdots + b_1 x + b_0$$

El polinomio producto

$$p(x) \cdot q(x) = r_{n+m}x^{n+m} + r_{n+m-1}x^{n+m-1} + \cdots + r_1 x + r_0$$

tiene por coeficientes:

$$r_k = \sum_{i+j=k} a_i b_j \qquad \text{con} \quad k = 0, 1, \ldots, n+m$$

Se pide que hagas un programa que calcule el producto de dos polinomios cuyos coeficientes se introducirán por el teclado y el polinomio producto resultante, se imprimirá en un fichero de datos, cuyo nombre se deberá introducir por teclado, previamente.

57. Realiza una tabla de números primos. Se debe introducir un número natural n y escribir en un fichero, la tabla de todos los primos menores o iguales que n, así como la cantidad de números primos que hay. Un ejemplo de ejecución sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 12345
**************************

Introduce un número entero positivo mayor que 6 -> 25
```

y el resultado

```
Hay 10 números primos menores o iguales que 25 y son:

1 ; 2 ; 3 ; 5 ; 7 ; 11 ; 13 ; 17 ; 19 ; 23
```

58. Dada una matriz $\mathbf{A} \in \mathcal{M}_{m\times n}$ haz un programa que imprima su matriz transpuesta en un fichero de datos. A continuación, si $\mathbf{A}$ es una matriz cuadrada de dimensión n, haz un programa que determine su transpuesta y la almacene en las mismas posiciones de memoria que la matriz de entrada (obviamente los datos de la matriz de entrada cambiarán por los coeficientes de la matriz transpuesta). Si quieres puedes utilizar el *dimensionamiento dinámico*.

59. ***Producto diádico o tensorial de dos vectores.***
A partir de dos vectores $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ se puede construir una matriz de la forma

$$[\boldsymbol{a} \otimes \boldsymbol{b}] = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} = \begin{bmatrix} a_1 b_1 & a_1 b_2 & \cdots & a_1 b_n \\ a_2 b_1 & a_2 b_2 & \cdots & a_2 b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_n b_1 & a_n b_2 & \cdots & a_n b_n \end{bmatrix} \tag{1}$$

En un espacio vectorial euclídeo (como $(\mathbb{R}^n, \cdot)$), esta operación se llama ***producto diádico o tensorial*** de $\boldsymbol{a}$ y $\boldsymbol{b}$ y da lugar a una *forma bilineal* que recibe el nombre de *tensor* y opera de la forma

$$(\boldsymbol{a} \otimes \boldsymbol{b}) \cdot \boldsymbol{v} = (\boldsymbol{b} \cdot \boldsymbol{v})\, \boldsymbol{a} \qquad \forall\, \boldsymbol{v} \in \mathbb{R}^n$$

Aplicando esta definición, se obtienen las componentes de este *producto diádico*

$$[\boldsymbol{a} \otimes \boldsymbol{b}]_{ij} = \boldsymbol{e}_i \cdot ((\boldsymbol{a} \otimes \boldsymbol{b}) \cdot \boldsymbol{e}_j) = \boldsymbol{e}_i \cdot ((\boldsymbol{b} \cdot \boldsymbol{e}_j)\, \boldsymbol{a}) = (\boldsymbol{b} \cdot \boldsymbol{e}_j)\, \boldsymbol{a} \cdot \boldsymbol{e}_i = a_i\, b_j$$

y que matricialmente coincide con la expresión (1).

Se pide que realices un programa, tal que dados dos vectores, calcule su *producto diádico* que se expresará matricialmente en un fichero de datos, cuyo nombre debe ser introducido desde el teclado.

60. Escribe un programa que calcule $\|\mathbf{A}\|_\infty$ y $\|\mathbf{A}\|_1$ con $\mathbf{A} \in \mathcal{M}_{n\times m}$.

$$\|\mathbf{A}\|_\infty = \max_{1\leqslant i\leqslant n} \sum_{j=1}^{m} |a_{i,j}| \qquad \|\mathbf{A}\|_1 = \max_{1\leqslant j\leqslant m} \sum_{i=1}^{n} |a_{i,j}|$$

61. Escribe un programa que determine si una matriz cuadrada es *diagonal dominante*. Se dice que una matriz cuadrada es *diagonal dominante* si

$$|a_{i,i}| \geqslant \sum_{\substack{j=1\\ i\neq j}}^{n} |a_{i,j}|$$

62. Resuelve la ecuación de segundo grado utilizando el *tipo* **`complex`**, para la solución.

63. Haz un programa que pida un número complejo por teclado y compruebe la igualdad:

$$e^{iz} = \cos z + i \text{ sen } z$$

64. ***Perímetro de un polígono.***
Un polígono $[p_1, \ldots, p_n]$ en el plano, con n lados y n vértices $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ es una *curva cerrada simple* que consiste en los segmentos que unen vértices consecutivos. Se ha de entender que p_1 es el vértice siguiente a p_n. Si se define

$$d(p_i, p_{i+1}) = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

entonces el perímetro del polígono es

$$\text{Perímetro}([p_1, \ldots, p_n]) = \sum_{i=1}^{n} d(p_i, p_{i+1})$$

entendiendo que el siguiente al índice n se toma como 1, en lugar de $n + 1$, como ya se ha comentado.

Escribe un programa que pida al usuario los vértices de un polígono plano y que calcule el perímetro del polígono introducido. Para hacer el ejercicio, se debe definir un vector de *tipo* **`punto`**, en la que en cada componente es una *estructura* que contiene las respectivas coordenadas de cada vértice del polígono. Un ejemplo de ejecución sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce el número de vértices ( > 1) -> 5

Introduce los vértices del polígono:
Introduce X[1] = 0
Introduce Y[1] = 0

Introduce X[2] = 2
Introduce Y[2] = 1

Introduce X[3] = 3
Introduce Y[3] = 4

Introduce X[4] = 0
Introduce Y[4] = 3

Introduce X[5] = 1
Introduce Y[5] = 2
```

y la salida, en un fichero, debe ser

```
Se ha introducido el polígono con vértices:
X[1] = 0.00  Y[1] = 0.00
X[2] = 2.00  Y[2] = 1.00
X[3] = 3.00  Y[3] = 4.00
X[4] = 0.00  Y[4] = 3.00
X[5] = 1.00  Y[5] = 2.00


El perímetro es 12.21090
```

65. Se pide que realices un programa que permita descomponer una matriz cuadrada real de tamaño n, $\mathbf{A}$ en suma de una matriz simétrica $\mathbf{S}$ y una matriz antisimétrica $\mathbf{T}$, utilizando la siguiente expresión:

$$\mathbf{A} = \underbrace{\frac{1}{2}\left(\mathbf{A} + \mathbf{A}^t\right)}_{\mathbf{S}} + \underbrace{\frac{1}{2}\left(\mathbf{A} - \mathbf{A}^t\right)}_{\mathbf{T}}$$

en donde $\mathbf{A}^t$ es la matriz traspuesta de $\mathbf{A}$. Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
*************************

Introduce el tamaño de la matriz -> 3

Introduce A[1][1] = 1
Introduce A[1][2] = 2
Introduce A[1][3] = 3

Introduce A[2][1] = 4
Introduce A[2][2] = 5
Introduce A[2][3] = 6

Introduce A[3][1] = 7
Introduce A[3][2] = 8
Introduce A[3][3] = 9
```

La salida típica en un fichero de datos debe ser:

```
La matriz introducida es:
A[1][1] = 1.00;  A[1][2] = 2.00;  A[1][3] = 3.00;
A[2][1] = 4.00;  A[2][2] = 5.00;  A[2][3] = 6.00;
A[3][1] = 7.00;  A[3][2] = 8.00;  A[3][3] = 9.00;

La matriz simétrica es:
S[1][1] = 1.00;  S[1][2] = 3.00;  S[1][3] = 5.00;
S[2][1] = 3.00;  S[2][2] = 5.00;  S[2][3] = 7.00;
S[3][1] = 5.00;  S[3][2] = 7.00;  S[3][3] = 9.00;


La matriz anti-simétrica es:
T[1][1] = 0.00;  T[1][2] = -1.00;  T[1][3] = -2.00;
T[2][1] = 1.00;  T[2][2] = 0.00;  T[2][3] = -1.00;
T[3][1] = 2.00;  T[3][2] = 1.00;  T[3][3] = 0.00;


La matriz suma debe coincidir con la inicial:
S[1][1] + T[1][1] = 1.00;  S[1][2] + T[1][2] = 2.00;  S[1][3] + T[1][3] = 3.00;
S[2][1] + T[2][1] = 4.00;  S[2][2] + T[2][2] = 5.00;  S[2][3] + T[2][3] = 6.00;
S[3][1] + T[3][1] = 7.00;  S[3][2] + T[3][2] = 8.00;  S[3][3] + T[3][3] = 9.00;
```

(Propuesto en examen: curso 2014/15).

Capítulo–6. Punteros

66. Realiza un programa que declare e inicialice una variable de *tipo* **`double`**. A continuación declara un *puntero a double*, que apunte a la variable anterior. A través del puntero, suma y resta una cantidad constante. En un fichero, escribe el valor inicial de la variable con su dirección y el resultado final con su dirección.

67. Dado el vector real $\mathbf{V} = [-2{,}45, 0{,}23, -45{,}98, -23{,}25, 1{,}89]$, se pide que realices un programa que asigne un puntero al vector y mediante este puntero, opera adecuadamente para incrementar cada componente en 25 unidades. En un fichero de salida se debe escribir el resultado final, con la dirección de memoria de cada componente. Primero hazlo con un *tipo* **`float`** y después con un *tipo* **`double`**. Observa el valor de las direcciones de memoria.

68. ***Ejemplo de*** **`*++ptr`**. Se pide que asignes el puntero **`ptr`** a la cadena de caracteres: **`Me gusta programar en C`** y a continuación imprime en un fichero, la salida como consecuencia de operar: **`*++ptr`**.

69. ***Ejemplo de*** **`(*ptr)---`**. Se pide que asignes el puntero **`ptr`** a la cadena de caracteres: **`Me gusta programar en C`** y a continuación imprime en un fichero, la frase escrita del revés.

70. Dada la matriz real

$$\mathbf{A} = \begin{bmatrix} -1{,}56 & 5{,}0 & -23{,}764 \\ 34{,}257 & 12{,}34 & -25{,}074 \\ 95{,}72 & -84{,}73 & 0{,}34 \end{bmatrix}$$

se pide que le asocies un puntero y a través de este puntero escribas los valores de las componentes junto con las respectivas direcciones de memoria.

71. Escribe un programa que permute las filas dadas de una matriz real dada.

72. En este ejercicio se trata de ilustrar la diferencia del concepto de matriz en **`C`**, con el de *doble puntero* (**`** **`PtrMat`**).

a) Declara e inicializa la matriz

$$\mathbf{A} = \begin{bmatrix} -1{,}56 & 5{,}0 & -23{,}764 \\ 34{,}257 & 12{,}34 & -25{,}074 \\ 95{,}72 & -84{,}73 & 0{,}34 \end{bmatrix}$$

con el tipo **`double`**.

b) Declara un *doble puntero a double*

c) Asigna la dirección de la matriz anterior a este puntero (lo más probable es que dé error).

d) Declara un *puntero matricial a double*. Asocia la dirección de la matriz a este puntero (ahora, no debería dar error).

La conclusión es que el *doble puntero* se puede utilizar para realizar una distribución matricial de datos, mediante la función **`malloc`**, pero **no** es una matriz desde el punto de vista *riguroso* de **`C`**.

73. Crea una matriz dinámicamente utilizando la función **`malloc`**, de manera que se pueda introducir la matriz

$$\mathbf{A} = \begin{bmatrix} -1{,}56 & 5{,}0 & -23{,}764 \\ 34{,}257 & 12{,}34 & -25{,}074 \\ 95{,}72 & -84{,}73 & 0{,}34 \end{bmatrix}$$

después escribe en un fichero esta matriz, con las direcciones de memoria de cada una de sus componentes.

Capítulo–7. Funciones

74. Haz un programa en el que se defina la función **`Presentacion()`** cuyo objetivo es que escriba en pantalla, el número de matrícula, el nombre y los apellidos. Mejora esta función para que los mismos datos se escriban en un fichero de salida.

75. A partir de la función **`potencia`**, que debes haber definido previamente, realiza un programa en la que se defina y utilice una función que calcule la potencia de dos números enteros. Esto es, dados m y n, que calcule mediante una función, el valor: m^n y lo imprima en un fichero.

76. Utiliza la función definida anteriormente para calcular las *variaciones con repetición de m elementos tomados de n en n* ya planteado en problemas anteriores.

77. Define una función **`Factorial`** y utilízala, para calcular las *variaciones sin repetición de m elementos tomados de n en n* y para determinar el número combinatorio

$$\binom{m}{n} = \frac{m!}{n!(m-n)!}$$

conceptos ya vistos en problemas anteriores. Una vez visto el concepto de *recursividad*, reescribe la función **`Factorial`** para que el cálculo se realice de forma recursiva.

78. Escribe un programa que dado un número $m \in \mathbb{N}$, calcule todos los números combinatorios siguientes:

$$\binom{m}{0}, \binom{m}{1}, \binom{m}{2}, \ldots, \binom{m}{m}$$

y muestre todos ellos en un fichero de salida, en el mismo orden. En este ejercicio y a partir del problema anterior, es obligado que se defina y utilice una función **`combinatorio`** cuyo argumento de entrada sean dos números positivos y la salida su número combinatorio correcto.
(Propuesto en examen: curso – 2014/15)

79. Se llama *binomio de Newton* a la igualdad:

$$(a+b)^m = \sum_{k=0}^{m} \binom{m}{k} a^{m-k}\, b^k$$

a) Se pide que se escriba un programa que dado un número natural $m \geqslant 1$ calcule 2^m, **SIN** utilizar la función `pow(x,y)` de la biblioteca de **C**. Por tanto, se tendrá que definir la función correspondiente.

b) Además el programa debe evaluar el *binomio de Newton* (esto es, programar el miembro de la derecha de la igualdad anterior), para comprobar que el cálculo anterior es correcto y mostrarlo en un fichero de salida. Después deben analizarse si ambos resultados son iguales. Si los son, la salida debe ser

`Se cumple la igualdad establecida por el Binomio de Newton`

si no,

`Uff!! algo falla en el programa`.

(Propuesto en examen: curso – 2014/15)

80. Escribe una función que describa el *Algoritmo de Horner* y utilízala para determinar el valor de un polinomio dado en el punto x_0.

81. Dado un vector real de doble precisión:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$$

se pide, que realices un programa que permita:

a) Calcular el valor de las normas:

$$\|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}\cdot\boldsymbol{x}}; \qquad \|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|; \qquad \|\boldsymbol{x}\|_\infty = \max_{1\leqslant i\leqslant n} |x_i|$$

b) Invertir las componentes del vector inicial.

c) Ordenar de menor a mayor, las componentes del vector inicial.

Es obligatorio que se definan y utilicen funciones para cada cuestión planteada. Por tanto debe haber 5 funciones: **`Norma_2`**, **`Norma_1`**, **`Norma_inf`**, **`Invierte`** y **`Ordena`**. Las salidas para las normas debe ser real, con al menos 13 cifras significativas. Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
*************************

Introduce la dimensión del vector -> -1
Por favor introduce un número positivo !!!! -> 5
Introduce V[1] = 0
Introduce V[2] = -1
Introduce V[3] = 1
Introduce V[4] = 3
Introduce V[5] = -6
```

La salida típica en un fichero de datos debe ser:

```
El vector introducido  es:
V[1] = 0.00; V[2] = -1.00; V[3] = 1.00; V[4] = 3.00; V[5] = -6.00;
La norma 2 es: 6.85565

La norma 1 es: 11.00000

La norma infinito es: 6.00000

El vector invertido es:
V[1] = -6.00; V[2] = 3.00; V[3] = 1.00; V[4] = -1.00; V[5] = 0.00;

El vector ordenado es:
V[1] = -6.00; V[2] = -1.00; V[3] = 0.00; V[4] = 1.00; V[5] = 3.00;
```

(Propuesto en examen: curso 2014/15)

82. Teniendo en cuenta la *tabla de números primos* realizada en un problema anterior, haz un programa que construya una tabla con la descomposición en factores primos de un número natural dado N. Para ello, debe construirse una función que determine los números primos menores que N y utilizar este resultado, para obtener la descomposición en factores primos.

83. Se llama *número de Armstrong* al número que verifica la propiedad de que *la suma de cada uno de sus dígitos elevado a la potencia del número de dígitos, coincide con su valor*. Por ejemplo el número 371 es un *número de Armstrong* ya que cumple

$$3^3 + 7^3 + 1^3 = 371$$

o bien, el número 92727

$$9^5 + 2^5 + 7^5 + 2^5 + 7^5 = 92727$$

Otros *números de Armstrong* son: $1, 153, 370, 371, 407, 1634, 8208, 9474, 54748, 92727, 93084, 548834$. Se pide que hagas una función de manera que dado un número natural N, determine si es de *Armstrong* o no. A partir de esta función, haz un programa que calcule los *números de Armstrong* menores de 10^8 (hay 27 y menores a 10^9, 31) y los imprima en un fichero.

84. Haz una función tal que dada una matriz permita permutar dos filas de la matriz.

85. Haz una función tal que dada una matriz $\mathbf{A} \in \mathcal{M}_{n\times m}(\mathbb{R})$, se obtenga su transpuesta.
(Sugerencia: constrúyase la matriz transpuesta sobre una matriz cuadrada de tamaño $\max(n, m)$).

86. Dada una matriz $\mathbf{A} \in \mathcal{M}_{n\times m}$, se dice que uno de sus coeficientes es un *punto de silla* si es el máximo de su fila y mínimo de su columna o bien, si es el mínimo de su fila y máximo de su columna. Se pide que hagas un programa que determine los posibles puntos de silla de una matriz entera dada $\mathbf{A} \in \mathcal{M}_{n\times m}(\mathbb{Z})$. Se deben definir y utilizar al menos 3 funciones: **`Maximo`**, **`Minimo`** y **`PuntoSilla`** desde la cual, se debe imprimir la solución. Una ejecución típica del programa debe ser:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce las filas de la matriz -> 3
Introduce las columnas de la matriz -> 3

Introduce A[1][1] = 1
Introduce A[1][2] = 2
Introduce A[1][3] = 3

Introduce A[2][1] = 5
Introduce A[2][2] = 6
Introduce A[2][3] = 7

Introduce A[3][1] = 4
Introduce A[3][2] = 1
Introduce A[3][3] = 9
```

La salida típica en un fichero de datos debe ser:

```
La matriz matriz introducida es
A[1][1] = 1;  A[1][2] = 2;  A[1][3] = 3;
A[2][1] = 5;  A[2][2] = 6;  A[2][3] = 7;
A[3][1] = 4;  A[3][2] = 1;  A[3][3] = 9;

El coeficiente A[1][3] = 3 es punto de silla.
El coeficiente A[2][1] = 5 es punto de silla.
```

(Propuesto en examen: junio – 2015).

87. Haz un programa que calcule el área de un polígono *simple*, mediante la utilización de una función **`area`**. Un argumento de entrada de la función debe ser un *puntero al vector* del *tipo* **`punto`**, que contiene las coordenadas de los vértices.

88. Haz un programa que calcule el *m.c.d.* de dos números naturales mediante el *algoritmo de Euclides*, definida en una función cuya codificación debe estar realizada de manera recursiva.

89. En un curso de álgebra lineal se define la *Traza de una matriz* $\boldsymbol{A} \in \mathcal{M}_{m\times n}$ como la suma de las componentes de la diagonal principal. Esto es

$$\boldsymbol{A} = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \dots & a_{m,n} \end{bmatrix} \Longrightarrow Tr(\boldsymbol{A}) = \sum_{i=1}^{k} a_{ii} \qquad \text{donde } k = \min(n, m)$$

En el caso de matrices del mismo tamaño, el concepto de *traza* se puede asociar al de un *operador lineal*.

En este ejercicio se pide que dada una matriz *simétrica aleatoria* $\boldsymbol{A} \in \mathcal{M}_{n\times n}$ calcules su *componente desviatoria* esto es

$$\boldsymbol{A}_D = \boldsymbol{A} - \frac{1}{n} Traza(\boldsymbol{A})\boldsymbol{I}$$

donde $\boldsymbol{I}$ es la matriz identidad. El concepto de *componente desviatoria* es una generalización matemática de la *componente desviatoria* del *tensor de tensiones*, que indica cuánto se aparta el estado tensional un sólido, respecto de un estado hidrostático o isotrópico.

Para realizar el ejercicio se debe:

a) definir una función llamada `MatSimAl` que obtenga la matriz simétrica aleatoria, en la cual se pida los extremos del intervalo en donde estarán los números aleatorios, así como la semilla para generarlos.

El proceso para generar esta matriz es muy sencillo. No obstante, se puede seguir la siguiente sugerencia

> Como argumento de la función debe estar el puntero a una matriz cuadrada. A continuación, en la parte triangular superior de la matriz se situarán los números aleatorios (el proceso es similar al realizado en clase de prácticas, para imprimir figuras triangulares de ejercicios planteados en el capítulo de *Sentencias de Control*). Es decir,
>
> $$a_{i,\,j} = \alpha \qquad \text{cumpliendo } i \leqslant j \text{ y donde } \alpha \text{ es un número aleatorio, que irá variando}$$
>
> Una vez asignados estos valores, se rellenan las componentes de la parte triangular inferior con la regla:
>
> $$a_{j,\,i} = a_{i,\,j}$$
>
> (ya planteado en el ejercicio 6 del capítulo de *Datos estructurados*).

b) definir una función llamada `Traza` que determine la traza de una matriz cualquiera.

c) definir una función llamada `MatDesv` que calcule la *componente desviatoria* de la matriz dada $\boldsymbol{A}$, quedando almacenada en la propia matriz $\boldsymbol{A}$.

d) escribir, desde el programa principal, la *componente desviatoria* en un fichero.

Una ejecución típica del programa con salida en pantalla sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce el nombre del fichero -> Salida.dat

Introduce el tamaño de la matriz ( > 1) -> -7
Introduce el tamaño de la matriz ( > 1) -> 3


Introduce los extremos del intervalo (enteros) -> 5 -1

Introduce la semilla -> -1

Se genera una matriz  de números aleatorios de tamaño 3 x 3

El intervalo es [-1, 5]
La semilla es -1

La matriz aleatoria simétrica es:
3.00;  -1.00;   2.00;
```

```
-1.00;   3.00;  -1.00;
2.00;  -1.00;   1.00;


La componente desviatoria  es:
0.67;  -1.00;   2.00;
-1.00;   0.67;  -1.00;
2.00;  -1.00;  -1.33;
```

(Propuesto en examen: curso 15/16).

90. En un curso de álgebra lineal se define la *Traza de una matriz* $\boldsymbol{A} \in \mathcal{M}_{m\times n}$ como la suma de las componentes de la diagonal principal. Esto es

$$\boldsymbol{A} = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \dots & a_{m,n} \end{bmatrix} \Longrightarrow Tr(\boldsymbol{A}) = \sum_{i=1}^{k} a_{ii} \qquad \text{donde } k = \text{mín}(n, m)$$

En el caso de matrices del mismo tamaño, el concepto de *traza* se puede asociar al de un *operador lineal*.

En este ejercicio se pide que dado un vector *aleatorio real* $\boldsymbol{v} \in \mathbb{R}^n$, se calcule el valor de la *traza* del producto diádico del vector $\boldsymbol{v}$ por sí mismo. Esto es

$$Tr(\boldsymbol{v} \otimes \boldsymbol{v})$$

Para ello se debe:

a) definir una función llamada `VectorAleatorio` que genere el vector aleatorio $\boldsymbol{v}$, en la cual se pida los extremos del intervalo en donde estarán los números aleatorios, así como la semilla para generarlos.

b) definir una función llamada `ProdTensorial`, que a partir del vector $\boldsymbol{v}$ se genere la matriz correspondiente al *producto diádico o tensorial* del vector $\boldsymbol{v}$ por sí mismo.

c) definir una función llamada `Traza` que determine la traza de una matriz cualquiera y sea utilizada para obtener el resultado pedido. Su aplicación debe hacerse desde el programa principal.

Una ejecución típica del programa con salida en pantalla sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
*************************

Introduce el nombre del fichero -> Salida.dat

Introduce la cantidad de números a generar ( > 0) -> -7
Introduce la cantidad de números a generar ( > 0) -> 5

Introduce los extremos del intervalo (enteros) -> 2 -1

Introduce la semilla -> -3

******** Salida de resultados ********

El intervalo es [-1, 2]
La semilla es -3
El vector aletorio es de tamaño 5 con valores:
-0.99744; -0.24540; -0.96219;  1.61400; -0.02228;
```

```
La matriz del producto diádico es:
0.99488;   0.24477;   0.95972;  -1.60986;   0.02222;
0.24477;   0.06022;   0.23612;  -0.39607;   0.00547;
0.95972;   0.23612;   0.92580;  -1.55297;   0.02144;
-1.60986;  -0.39607;  -1.55297;   2.60500;  -0.03596;
0.02222;   0.00547;   0.02144;  -0.03596;   0.00050;


El valor de la traza es 4.58640
```

(Propuesto en examen: curso 15/16).

91. Programa un código que calcule áreas y perímetros de figuras planas. Se ofrecerá al usuario un menú con la posibilidad de elegir entre cuatro opciones y una vez que el usuario haya elegido una figura, se le pedirá que introduzca los datos necesarios para los cálculos.

Para cada figura deberá construirse una función que calcule su área y su perímetro en términos de los datos introducidos.

Las coordenadas de los vértices y las salidas de las funciones se guardarán en variables de tipo estructura.

Figuras a ofrecer y datos a pedir:

- Círculo: radio.
- Cuadrado: vértices.
- Triángulo: vértices.

Concluido el programa y comprobado su buen funcionamiento, se añadirá una función que guarde en un fichero los datos y los resultados. El nombre del fichero será introducido por teclado desde la función principal.

Se ofrecerá al usuario la oportunidad de elegir más de una figura. En ese caso, todos los resultados se guardarán en el mismo fichero.

(Propuesto en examen: curso 15/16).

92. Programa un código para calcular el área y el perímetro de un polígono simple cuyos vértices son puntos del plano real. Para el cálculo del perímetro debe usarse la fórmula de la distancia entre dos puntos y para el cálculo del área la expresión

$$\text{Área} = \frac{1}{2}\left|\sum_{i=0}^{n-1} \det\begin{pmatrix} x_i & x_{i+1} \\ y_i & y_{i+1} \end{pmatrix}\right|,$$

donde n es el número de vértices y (x_n, y_n) son las coordenadas del primer punto.

- El número de vértices será introducido por teclado, con control de dato correcto.
- Las coordenadas de los vértices se introducirán por teclado y se guardarán en variables de tipo estructura. Es necesario que los vértices se introduzcan de forma consecutiva, siguiendo el borde del polígono.
- Deberá haber al menos cuatro funciones, cada una con uno de los siguientes objetivos:
  - Cálculo de la distancia entre dos vértices.
  - Cálculo del determinante para cada par de vértices.
  - Cálculo del perímetro.
  - Cálculo del área.
- Concluido el programa y comprobado su buen funcionamiento, se añadirá una función que imprima en un fichero los datos y los resultados. El nombre del fichero será introducido por teclado desde la función principal.

- Se ofrecerá al usuario la oportunidad de introducir más de un polígono. En ese caso, todos los resultados se guardarán en el mismo fichero.

(Propuesto en examen: curso 15/16).

93. Dado un PVI (problema de valor inicial)

$$y' = e^{-y^2} P(x), \qquad y_0 = y(x_0) = y(0) = P(0)$$

siendo $P(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \cdots + a_n$ un polinomio de grado n con coeficientes enteros. Se desea aproximar el valor de la solución $y(x)$ de la edo en el punto $x = 1$ mediante la sucesión definida por

$$y_{k+1} = y_k + \frac{e^{-y_k^2} P(x_k)}{M} \quad k = 0, 1, \ldots, M-1, \quad \text{siendo } x_i = \frac{i}{M} \qquad i = 0, 1, \ldots, M$$

En este ejercicio se pide que dado un polinomio con coeficientes enteros $P(x)$ obtengas la sucesión que genera el método para aproximar $y(1) \approx y_{\text{M}}$.

Para realizar el ejercicio se debe:

a) introducir desde teclado el nombre de un fichero en el que se escribirán los datos de salida.

b) introducir el número de pasos, M, para generar la sucesión que aproxima $y(1)$, así como el grado del polinomio $P(x)$, n.

c) definir una función llamada `GenPol` que obtenga los coeficientes enteros aleatorios del polinomio $P(x)$, en la cual se pida los extremos del intervalo en donde estarán los números aleatorios, así como la semilla para generarlos. Los coeficientes del polinomio resultante serán **reales de doble precisión**, se imprimirán por pantalla y en el fichero desde el programa principal.

d) definir una función llamada `Pol` que calcule, y devuelva, el valor del polinomio $P(x)$ para un valor real cualquiera x, que deberá ser **real de doble precisión**.

e) definir una función llamada `Euler` que genere la sucesión de valores $y_i \quad i = 0, 1, \ldots, M$ que será devuelta al programa principal en un vector.

f) por último, desde el programa principal debe imprimirse en el fichero y por pantalla la sucesión de puntos $(x_i, y_i) \quad i = 0, 1, 2, \ldots, M$, siendo $y_{\text{M}} \approx y(1)$

Una ejecución típica del programa con salida en pantalla sería:

```
***** Nombre: Jdkdkdkd
****  Apellidos: Skdkdkd Rdkdkdkd
**** Matricula: 1111111
**************************

Introduce el nombre del fichero -> Salida.dat

Introduce el grado del polinomio P(x), n>0 -> -7
Introduce el grado del polinomio P(x), n>0 -> 3

Introduce el número de pasos M (entero positivo) a realizar para aproximar y(1)

Introduce los extremos del intervalo (enteros) -> 5 -2

Introduce la semilla -> -1

******** Salida de resultados ********
```

```
Coeficientes (vector V ) generados->
coef.de grado 0-> V[0]=1.00  coef.de grado 1-> V[1]=1.00
coef.de grado 2-> V[2]=4.00  coef.de grado 3-> V[3]=3.00

Sucesión generada:
Salida de los puntos x(i) y sus aproximaciones y(i)
i    x(i)      y(i)
0    0.0000    1.0000
1    0.1000    1.0368
2    0.2000    1.0758
3    0.3000    1.1193
4    0.4000    1.1690
5    0.5000    1.2259
6    0.6000    1.2899
7    0.7000    1.3598
8    0.8000    1.4336
9    0.9000    1.5091
10   1.0000    1.5842
```

Propuesto en examen: junio – 2016).

94. Se pide que realices un programa que determine el vector $\boldsymbol{y} \in \mathbb{R}^n$, resultado del producto de la matriz $\mathbf{A} \in \mathcal{M}_{n\times n}$ con el vector $\boldsymbol{x} \in \mathbb{R}^n$. Matricialmente:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \dots & a_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \iff y_i = \sum_{j=1}^{n} a_{i,j}\, x_j \qquad \text{con} \quad i = 1, 2, \dots, n$$

Para ello se debe:

a) definir una función llamada **`VecMatInt`** en la que se establezcan las componentes del vector $\boldsymbol{x}$ y de la matriz cuadrada $\mathbf{A}$. Si la introducción se realiza mediante números aleatorios, en esta función se debe introducir el intervalo en donde estarán los números aleatorios, así como la semilla. Si se opta por introducir las componentes por teclado, deberá programarse de manera clara y ordenada para que el usuario sepa qué componentes del vector $\boldsymbol{x}$ y de la matriz cuadrada $\mathbf{A}$ está introduciendo.

b) definir una función llamada **`Prod`** en la que se realice el producto pedido.

c) imprimir, desde el programa principal, el vector $\boldsymbol{y}$ resultado del producto obtenido en la función **`Prod`**.

Una ejecución típica del programa con salida en pantalla sería:

```
Introduce el nombre del fichero -> Salida.dat

Introduce la dimension del vector y de la matriz ( > 0) -> -7
Introduce la dimension del vector y de la matriz ( > 0)  -> 3

Introduce los extremos del intervalo (enteros) -> 5 -2

Introduce la semilla -> -1
```

La salida en el fichero debe ser

```
***** Salida de resultados *******

**** Nombre: Jalslsl
```

```
**** Apellidos: Kskdiekdkm Rosldoe
**** Número de matrícula: 111111
**********************************

El intervalo es [-2, 5]
La semilla es -1
El vector aletorio es de tamaño 3 con valores:
1.00;  1.00;  4.00;

La matriz aleatoria es:
 3.00   2.00   4.00
 3.00   3.00   0.00
-2.00  -1.00  -2.00


El vector producto es:
21.00;  6.00; -11.00;
```

Propuesto en examen: julio – 2016).