Ejercicios para el Manejo Básico de Matlab/Octave

Gustavo Rivas Gervilla

17 de diciembre de 2019

¿Cómo usar este archivo?[1]

En este archivo puedes encontrar varios ejercicios de programación con Matlab\Octave . Este archivo es **un complemento** para el estudio de la asignatura Fundamentos de Informática del Grado en Ingeniería Química de la Universidad de Granada.

En cualquier caso no es una guía de estudio, o una muestra de todo aquello que el estudiante ha de poner en práctica en los exámenes de la asignatura. Es sólo un material adicional con el que poner en práctica distintos conceptos de la asignatura.

El archivo está dividido en 3 partes:

1. **Enunciados:** Aquí se encontraran los distintos ejercicios prácticos que el estudiante puede realizar para aumentar su destreza con la programación en Matlab\Octave .

2. **Recetas:** Para cada uno de los ejercicios se facilita un desglose del problema que plantea el ejercicio, para orientar al estudiante en escritura del código que resuelve el problema planteado.

3. **Soluciones:** Para cada ejercicio se proporciona un posible código para solucionar el problema.

El estudiante abordará cada uno de los ejercicios. En caso de no saber cómo abordar el ejercicio o no estar seguro de qué se pide en el enunciado puede consultar la *receta* facilitada para ese ejercicio. Finalmente, el estudiante puede consultar la solución propuesta para el ejercicio si quiere compararla con la suya, o ver en qué se está pudiendo equivocar.

Además:

- Este archivo ha sido creado empleando LaTeX.

- Los trozos de código que aparecen en el mismo han sido insertados y formateados empleando el paquete listings, con lo que el código se puede copiar directamente del PDF y pegarlo en un editor de código.

[1] Para cualquier duda sobre los ejercicios de este guion enviar un correo a griger@decsai.ugr.es

Índice

1. Enunciados

Enunciado. Ejercicio 1

Dado el siguiente vector con los radios de los círculos se pide:

- Mostrar el número de círculos que hay.
- Obtener un vector con el área de cada círculo. Recordemos que la fórmula es πr^2, donde r es el radio.
- Obtener un vector con la longitud de la circunferencia de cada círculo. La fórmula para este cálculo es $2\pi r$.

El vector con los radios de ejemplo es el siguiente. Aunque se puede probar con otros vectores de distintas longitud o valores:

```
v = [10 5 8 6 20 50];
```

Enunciado. Ejercicio 2

La siguiente matriz muestra la distancia entre diversas ciudades:

	A	B	C	D	E
A	0	10	20	30	40
B	15	0	20	50	60
C	20	30	0	80	10
D	50	40	20	0	10
E	20	20	54	21	0

Observa que la matriz no es simétrica, la ruta de A a B puede ser más larga o más corta que la ruta de B a A. Dada esta matriz:

- Comprueba que la distancia de cada ciudad a sí misma. Es decir, ¿es la traza de la matriz igual a cero?
- Calcula la distancia máxima entre dos ciudades.
- Calcula la media de la distancia de las rutas que parten de la ciudad A.
- Calcula la media de la distancia de las rutas que llegan a la ciudad C.

Enunciado. Ejercicio 3

La siguiente tabla muestra la nota de algunos estudiantes en diversas asignaturas sí:

	Matemáticas	Lengua	Filosofía
Estudiante1	5	7	10
Estudiante2	6	4	3
Estudiante3	8	5	8
Estudiante4	9	8	9

- Calcula la desviación típica en la nota para cada una de las asignaturas.
- Muestra la nota del Estudiante2 en Lengua.
- Calcula la media de cada estudiante redondeándola al alza.
- Muestra las notas de cada estudiante en un gráfico de barras.

Enunciado. Ejercicio 4

La altura de una pelota de goma que se deja caer al suelo desde una cierta altura viene dada por la siguiente función, donde t es el tiempo en segundos:

$$| \; sin(t^t)/2^{(t^t - \frac{\pi}{2})/\pi} \; |$$

Pide al usuario un instante de tiempo entre el 1 y 3, y calcula la altura de la pelota en ese instante. A continuación visualiza gráficamente cómo evoluciona la altura de la pelota en ese rango de tiempo, y testea mirando la gráfica si el valor obtenido se corresponde con la realidad.

Enunciado. Ejercicio 5

Dadas dos matrices 5x12 con elementos aleatorios en el intervalo [0,1000]. Calcular, suponiendo que la primera matriz representa los gastos de 5 personas en cada uno de los meses del año, y la segunda representa sus ingresos:

1. Para cada persona y cada mes la cantidad (negativa o positiva) neta que ha entrado en su cuenta en ese mes ($ingresos - gastos$).
2. El balance anual para cada persona (la suma de los ingresos en cada mes para cada persona).
3. El neto máximo mensual de cada persona. Y el neto mínimo.
4. La media de gastos anual entre todas las personas.
5. El ratio ingresos/gastos mensual de cada persona.

Enunciado. Ejercicio 6

Un estudiante ha obtenido las siguientes calificaciones en sus últimos exámenes: 5,6,8,4,3,8,7. Solicita al estudiante su nombre y muestra por pantalla:

1. La medida de sus notas.
2. Su mejor y peor nota.
3. Una gráfica mostrando la evolución de sus notas.

Estos resultados se mostrarán siempre haciendo mención expresa al nombre del usuario. Por ejemplo, "Fulanito, la media de tus notas es...".

Enunciado. Ejercicio 7

El número π se puede aproximar empleando el conocido Producto de Wallis el cuál nos dice que:

$$\frac{\pi}{2} = \prod_{n=1}^{\infty} \frac{4n^2}{4n^2+1}$$

Para comporobar que efectivamente sucesión converge a π:

- Calcula para cada natural m desde 1 hasta 1000, el valor del productorio $2\prod_{n=1}^{m} \frac{4n^2}{4n^2+1}$. Y observa cómo los valores se aproximan a π. Es importante que no metas el 2 dentro del productorio, es decir, primero se calcula el productorio y luego se multiplica por dos su valor. A diferencia de lo que ocurre con una sumatoria donde sí se podría meter dentro de la sumatoria algo que va multiplicándola.
- Muestra una gráfica donde se muestren esos valores calculados y observa como los valores convergen a π.

Enunciado. Ejercicio 8

Dado el siguiente vector

```
v = [-1 5 6 8 9 10 20 -5 6 8];
```

Realiza las siguientes operaciones sobre este vector:

1. Muestra el vector en orden inverso.
2. Muestra el vector ordenado de menor a mayor.
3. Poner en valor absoluto los elementos en las posiciones impares.
4. Duplicar los elementos en las posiciones múltiplo de 3.
5. Obtener la suma de los 5 primeros elementos.
6. Obtener el producto de los elementos en posiciones pares.
7. Restar 1 a los 5 últimos elementos del vector.

Enunciado. Ejercicio 9

Un cuadrado mágico es una matriz cuadrada de un tamaño determinado, en la que aparecen todos los números de 1 hasta n^2 sin repetir, siendo n el tamaño de la matriz. Además esta matriz cumple que la suma de los elementos en cada fila, la suma de los elementos de cada columna y la suma de los elementos de las dos diagonales principales de la matriz es la misma.

La función `magic` en Matlab nos genera un cuadrado mágico con el tamaño que nosotros le pasemos como argumento.

Genera una cuadrado mágico con dicha función y comprueba que sus filas, columnas y diagonales suman todas lo mismo.

Enunciado. Ejercicio 10

Dada una matriz (cualquiera, la puedes definir como quieras) obten la matriz resultante de multiplicar los elementos de cada fila de la matriz original por el primer elemento de esa fila.

Enunciado. Ejercicio 11

Solicitar al usuario su nombre y su saldo actual. Supuesto que un lápiz cuesta 1.5€ y una goma de borrar cuesta 2€. Calcula:

- Cuál es el número máximo de lápices que puede comprar, y cuántas gomas puede comprar con el dinero sobrante.
- Y viceversa: cuál es el número máximo de gomas que puede comprar, y cuántos lápices puede comprar con el dinero sobrante.

Dirígete al usuario empleando su nombre.

Enunciado. Ejercicio 12

La siguiente igualdad es una generalización de la identidad de Euler, donde n es un número natural mayor que 1, e i es la unidad imaginaria:

$$\sum_{k=0}^{n-1} e^{2\pi i \frac{k}{n}} = 0$$

Comprueba que esta identidad se da para $n = 100, 1000$ y 10000.

Enunciado. Ejercicio 13

Para ciertos ángulos el seno y el coseno coinciden. Esto se produce cuando el seno y el coseno valen $\pm\frac{\sqrt{2}}{2}$. Para comprobarlo dibuja las funciones seno y coseno en el intervalo $[0, 10\pi]$, y añade también dos líneas horizontales a las alturas $\frac{\sqrt{2}}{2}$ y $-\frac{\sqrt{2}}{2}$, dándole distinto color y estilo de línea a cada una. Además, añade una leyenda que permita identificar el seno y el coseno en la gráfica.

Enunciado. Ejercicio 14

Para repasar la preferencia de operadoradores en Matlab, usa una calculadora si la necesitas para obtener el resultado de las siguientes operaciones y comprueba en Matlab si el resultado es el esperado:

1. $\sqrt{2^{10}} + e^{10/5+4} log(5)$
2. $log_{10}(100)(\sqrt[4]{128} + 50)$
3. $sin(log(20))csc(\pi/2)$
4. $(asin(1) - cot(\frac{3\pi}{2}) + cos(\pi/8))+ \mid 5 - 8 \mid log(30)$

Enunciado. Ejercicio 15

Dada una matriz 3x3 (que la podemos definir como queramos)

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Calcular su determinante haciendo uso de la siguiente fórmula:

$$det(A) = a(ei - fh) - b(di - fg) + c(dh - eg)$$

Comprueba si el resultado es correcto empleando la función `det`.

Enunciado. Ejercicio 16

La siguiente matriz refleja el número de unidades que distintos clientes han comprado de un determinado producto:

	Producto A	Producto B	Producto C	Producto D
Cliente 1	100	200	50	40
Cliente 2	50	80	10	5
Cliente 3	50	50	60	210

Por otro lado el siguiente vector nos da el precio de cada producto:

Producto A	Producto B	Producto C	Producto D
50	20	10	30

Calcula:

1. Qué ha gastado cada cliente en total.
2. Qué han gastado los clientes en media en el Producto B.
3. El total gastado por todos los clientes en la tienda.

Enunciado. Ejercicio 17

Como sabemos, existen algunas matrices que no tienen una matriz inversa, es decir, una matriz que al multiplicarla por ella dé la identidad. Para ver esto define una matriz cuadrada de un tamaño cualquiera con todos sus elementos nulos, y otra con todos sus elementos uno. Y comprueba el resultado que Matlab\Octave devuelve al intentar calcular su matriz inversa.

Enunciado. Ejercicio 18

El siguiente vector recoge la altura de 10 personas:

1.5 1.9 1.35 2 1.6 1.72 1.7 1.85 1.4 1.45

Calcular:

- La altura de la persona más alta y de la más baja.
- La media de los individuos que ocupan una posición múltiplo de tres.
- La media de las 5 personas más bajas.
- La suma de la altura de las 3 personas más altas.

Enunciado. Ejercicio 19

Genera un vector de 100 números aleatorios en el rango $[5, 10]$ y dibuja los valores obtenidos. Para ello recuerda la siguiente formula cambia la escala, generando para cada número x en un rango $[a, b]$, su correspondiente número en el rango $[c, d]$:

$$x \longrightarrow c + \frac{(d-c)(x-a)}{b-a}$$

Enunciado. Ejercicio 20

Una empresa cuenta con varios empleados con los siguientes salarios:

1100 2500 1350 2000 2000 1800

Baraja los empleados aleatoriamente y:

- Aplica un aumento de un 10 % a los empleados que hayan quedado en posiciones pares.
- Aumenta 100€ a los empleados en las 3 primeras posiciones.
- Duplica el salario al empleado en la última posición.

Enunciado. Ejercicio 21

Obten una matriz en la que:

- La primera fila contenga valores de 1 a 10 con saltos de 0.2.
- La segunda contenga el valores de la función $\sqrt{x}$ sobre los valores de la primera fila.
- La tercera contenga los valores de la función $sin(x) * cos(x)$ sobre los valores de la primera fila.

- La cuarta fila contenga los valores de la función $log(x)$ sobre los valores de la primera fila.

Dibuja la gráfica de las 3 funciones empleando los datos contenidos en la matriz.

Enunciado. Ejercicio 22

Define una matriz aleatoria de dimensiones 5*10 con valores en el intervalo $[0, 1]$:

- Obten la suma de los elementos en las filas pares y columnas múltiplo de 3.
- Obten la suma de los elementos en las filas 1, 2 y 5, que estén en columnas pares.
- Obten una matriz en la que los elementos sean el cubo de los elementos de la matriz original.

Enunciado. Ejercicio 23

Dadas las dos matrices siguientes:

$$A = \begin{pmatrix} 2 & 3 & -5 \\ 10 & 1 & 0 \\ 5 & 5 & 6 \end{pmatrix} B = \begin{pmatrix} 4 & 0 & 1 \\ -9 & -2 & 5 \\ 0 & 1 & 7 \end{pmatrix}$$

Calcula:

- El producto de A por B, y de B por A.
- La matriz resultante de multiplicar las matrices elemento a elemento.
- $A * A * B$, o lo que es lo mismo A^2B
- El producto de A por su matriz inversa.
- La suma de las dos matrices.
- La matriz resultante de multiplicar los elementos de B por el determinante de A.
- La matriz resultante de dividir los elementos de A por la traza de B.

Enunciado. Ejercicio 24

En Matlab\Octave existen dos funciones que nos generan números aleatorios en el intervalo $[0, 1]$. `rand` genera valores en el este intervalo siguiendo una distribución uniforme, es decir, todos los valores tienen la misma probabilidad de salir. Por otro lado `randn` genera valores siguiendo una distribución normal de media 0 y varianza 1, donde los valores más probables son aquellos más cercanos a la media, más cercanos a cero (puesto que esta distribución tiene la forma de lo que se conoce como una campana de Gauss centrada en el cero):

- Obten un vector de 1000 números aleatorios con cada una de estas funciones.
- Calcula la media de cada uno de los vectores.

Verás como la media del vector generado con `rand` está cercana a 0,5; se obtienen valores distribuidos uniformemente a lo largo de todo el intervalo $[0, 1]$, cuyo punto medio es el 0,5. Por otro lado la media del vector obtenido con `randn` estará cercana a 0, que como hemos dicho anteriormente es la media que tiene la distrubución que siguen los valores generados con esta función. Puedes repetir el experimento tantas veces como desees.

Enunciado. Ejercicio 25

La siguiente matriz muestra el precio base de distintos productos en distintos países:

	Producto A	Producto B	Producto C
País 1	100	200	300
País 2	400	300	200
País 3	500	400	100

Y la siguiente matriz recoge el factor a aplicar a cada uno de esos productos en cada país debido a distintas políticas internacionales:

	Producto A	Producto B	Producto C
País 1	1.1	0.5	0.3
País 2	0.5	2	1.2
País 3	1	1	1.5

Calcula:

- El precio medio del Producto A en los distintos países después de aplicar los factores.
- La diferencia media absoluta en el precio del Producto B antes y después de aplicar los factores
- El precio medio de los productos en el País 2 después de aplicar los factores.
- La diferencia media absoluta de precios entre los países 1 y 3 después de aplicar los factores.

Enunciado. Ejercicio 26

Dado un vector con los valores del 1 al 100, escribe un script en el que:

- Con estructuras repetitivas cambiar el signo de los múltiplos de 5.
- Sin emplear estructuras repetitivas, empleando operadores relacionales, multiplicar por 2 los múltiplos de 3.

Enunciado. Ejercicio 27

Escribe una función que, dada una matriz devuelva:

- El número de elementos que son múltiplos de 2 o múltiplos de 3 (pero no de ambos), sin emplear estructuras repetitivas, es decir, empleando operadores relacionales.
- El número de elementos que son múltiplos de 5 y de 4, empleando estructuras repetitivas.

Enunciado. Ejercicio 28

Dado un vector de enteros aleatorios en el intervalo [-100,100]:

- Se imprime si el número de elementos positivos es mayor que el de negativos, igual o menor. Entiendo por elementos positivos aquellos que son mayor o igual que cero.
- Si el número de positivos es mayor que el de negativos: se imprime si en el vector hay más valores pares que impares, o no.
- Si el número de positivos es menor que el de negativos: se imprime el número de múltiplos de 3, y además se indica el caso en el que la suma de elementos múltiplos de 5 en valor absoluto, es mayor que la suma de elementos positivos.
- En caso de empate, los números negativos pasan a ser cero. Y los positivos no nulos se cambian de signo.
- En cualquier caso se imprime el número de valores nulos del vector.

Enunciado. Ejercicio 29

Define dos funciones `maximum` y `minimum` que calculen, empleando estructuras repetitivas el máximo y el mínimo de un vector.

Emplea estas dos funciones en un script en el que se defina un vector a mano de números positivos. Luego se pedirá al usuario números mientras inserte números mayores o iguales que cero. Cuando el usuario deje de introducir números se mostrará la suma de aquellos números introducidos por el usuario que estan en el intervalo [mínimo, máximo]. Donde mínimo y máximo son el mínimo y el máximo del vector previamente definido.

Enunciado. Ejercicio 30

Implementa una función que calcule empleando estructuras repetitivas la media de los elementos de la triangular superior de una matriz cuadrada.

Enunciado. Ejercicio 31

Implementa una función `esPrimo` que, empleando estructuras repetitivas, devuelva un booleano indicando si un número es primo o no.

A continuación, implementa un script en el que, dado un número introducido por el usuario indique:

- Si es un número par y mayor que 10, o no.
- En caso de serlo, se muestra si es un múltiplo de 5 o no.
- Si no lo es, se imprime si es un número primo o no, empleando la función anteriormente implementada.

Enunciado. Ejercicio 32

Implementa una función que dada un vector de números enteros positivos, dibuje un gráfico de barras usando asteriscos para dibujarlo, donde cada barra tiene la altura, es decir, tantos asteriscos, como indique el número en la posición del vector correspondiente.

Enunciado. Ejercicio 33

Escribe un script en el cual se pida el número de jugadores. Entonces se generará una cantidad máxima aleatoria en el intervalo [0,1000]. Para cada jugador se le indicará su cantidad actual, que empezará siendo cero, y le preguntará si deseada seguir obteniendo números o no. A lo que responderá "S" o "N". Cada vez que el usuario diga que "S" se le generará otro número aleatorio en el intervalo [0, cantidadMaxima / 10], que se sumará a su cantidad total.

Cuando el jugador diga "N" se plantará y pasará al siguiente, mostrando la cantidad total obtenida por este jugador. Se puede usar este script como un juego en el que ganará quien más se aproxime a la cantidad máxima sin pasarse.

Enunciado. Ejercicio 34

Implementa una función que empleando estructuras repetitivas calcule la traza de una matriz no necesariamente cuadrada.

Enunciado. Ejercicio 35

Implementa una función que genera un entero aleatorio entre el 0 y el 100.

Enunciado. Ejercicio 36

Implementa una función que devuelva el máximo de un vector, junto con la posición del vector en la que se encuentra. Emplea para ello estructuras repetitivas. A continuación emplea esta función en un script para mostrar el máximo de un vector, y además cambiar todas las posiciones del vector, excepto la perteneciente al máximo, por cero.

Enunciado. Ejercicio 37

Implementa un script en el que se le pida al usuario su nombre, su edad, y su peso. Entonces:

- A los menores de edad se les solicitará su altura en metros, y se mostrará su IMC ($peso/altura^2$).
- A los usuarios de entre 30 y 50 años se les solicitará su sueldo.
- A aquellos que tienen menos de 18 años se les preguntará si desean estudiar un idioma o no. Deberán responder "N" o "S", y en base a esta respuesta se les sugerirá que estudien sueco o que elijan música como actividad extraescolar.
- Dependiendo del sueldo del usuario, se les recomendará alquilar un piso, comprar un piso o comprar una casa:
  - Menos de 700€: alquilar piso.
  - Entre 700€ y 1500€: comprar piso.
  - Más de 1500€: comprar casa.
- Para aquellas personas mayores de 50 años se les sugerirá un plan de pensiones. Además si su peso excede los 100kg, entonces se les recomendará hacer deporte.

La información que se muestra por pantalla irá acompañada por el nombre del usuario.

Enunciado. Ejercicio 38

Escribe un script en el que dada una matriz, se cuente cuántos elementos son múltiplo de 2, o bien son múltiplos de 3 o de 5, pero no de ambos.

A continuación, empleando estructuras repetitivas, modifique la matriz de modo que los elementos de cada fila queden multiplicados por el primer elemento de esa fila.

Por último selecciona los elementos de la matriz múltiplos de 4.

Enunciado. Ejercicio 39

Escriba un script en el que empleando estructuras repetitivas se trasponga una matriz cuadrada. Para ello tenga en cuenta que sólo se tendrán que recorrer los elementos de la diagonal superior o inferior de la matriz, sin incluir la diagonal que, como sabemos, al transponer una matriz queda como estaba.

Enunciado. Ejercicio 40

Implemente una función que dados dos enteros positivos, devuelva el cociente y el resto de la división entera del primero por el segundo. Para ello se empleará el algoritmo de la división, que se describe como sigue:

1. El cociente comienza siendo 0, y el resto comienza siendo el dividendo de la división a realizar.

2. Mientras que el resto sea mayor o igual que el divisor.

    a) El cociente aumenta en uno.

    b) Al resto se le resta el divisor.

2. Recetas

Receta. Ejercicio 1

1. Declaramos el vector con los radios.

2. Mostramos el número de círculos por pantalla, empleando la función que nos da el número de elementos en un vector.

3. Usamos la fórmula para calcular el área con el vector de radios, aprovechando cómo funcionan las operaciones con vectores en Matlab.

4. Hacemos lo propio para calcular la longitud de la circunferencia de cada círculo.

Receta. Ejercicio 2

1. Definimos la matriz.

2. Mostramos por pantalla la traza de la matriz.

3. Mostramos el máximo de la matriz. Para ello tendremos que calcular el máximo dos veces, ya que la primera nos da el máximo de cada columna, puesto que calcula el máximo **a lo largo de** la primera dimensión.

4. Mostramos el máximo de la primera fila.

5. Mostramos el máximo de la tercera columna.

Receta. Ejercicio 3

1. Definimos la matriz.
2. Mostramos la nota del Estudiante2.
3. Calculamos la desviación típica en cada columna, es decir, a lo largo de la primera dimensión (a lo largo de las filas). Y mostramos el vector resultante.
4. Calculamos la media en cada fila, es decir, a lo largo de la segunda dimensión (se deja fija la fila y se mueve la columna). Y mostramos el vector resultante.
5. Mostramos la matriz empleando un gráfico de barras y añadimos título, etiquetas y leyenda para hacerlo más legible.

Receta. Ejercicio 4

1. Pedir al usuario el instante de tiempo t en el que calcular la altura, y hacer los cálculos necesarios para obtener la altura de la pelota en dicho instante.
2. Mostrar el resultado por pantalla.
3. Construir un vector que contenga el rango de valores donde se va a evaluar la función para dibujarla (los valores del eje x). Emplear saltos suficientemente pequeños en dicho rango para que la función se dibujo con la suficiente "resolución".
4. Obtener el vector de valores y a partir del vector anterior.
5. Dibujar la gráfica usando ambos vectores.

Receta. Ejercicio 5

1. Generar las matrices aleatorias, que tendrán valores en el rango $[0, 1]$. Ahora para tener valores en el rango $[0, 1000]$, no tenemos más que multiplicar esos elementos por 1000.
2. Calcular la diferencia de las dos matrices para obtener una matriz con el neto de cada mes para cada persona.
3. Hacer la suma por filas de la matriz anterior para obtener el neto anual, es decir, a lo largo de la segunda dimensión.
4. Sobre la matriz del segundo punto obtener el máximo y el mínimo de la fila, es decir, a lo largo de la segunda dimensión.
5. Obtener el gasto anual de cada persona y hacer la media de los valores obtenidos.

6. Hacer la división de la matriz de ingresos por la matriz de gastos, elemento a elemento.

Receta. Ejercicio 6

1. Definimos un vector que contenga las notas del estudiante.
2. Le pedimos al usuario su nombre. Teniendo en cuenta que la información introducida por el usuario ha de tratarse de forma literal y no evaluarse (recuerda la opción de `input` que nos permite esto).
3. Mostramos por pantalla la media, el mínimo y el máximo de ese vector.
4. Dibujamos una gráfica con el valor de sus notas.

Receta. Ejercicio 7

1. Definimos un vector con los números del 1 al 1000.
2. Para cada valor n en dicho vector calculamos el correspondiente término del productorio.
3. Ahora sobre ese nuevo vector con los términos calculamos otro vector con el producto acumulado. Así cada término se corresponderá con un productorio desde 1 hasta un cierto n.
4. Y multiplicamos ese vector por dos. Veremos cómo cada término se acerca más a π, es decir, cuántos más términos multiplicamos más nos acercamos a π.
5. Dibujamos los valores de este último vector en una gráfica.

Receta. Ejercicio 8

Para este ejercicio sólo conviene apuntar el modo más adecuado de obtener los m últimos elementos de un vector (siempre que el vector tenga al menos m elementos). Como sabemos `end` nos devuelve la última posición del vector, es decir, si un vector tiene 10 elementos, 10 posiciones, `end` valdrá 10. Por otro lado si el vector tiene 15 elementos `end` vale 15. Entonces la posición `end - h` sería la posición `h` posiciones antes que la última posición. Por tanto los últimos 5 elementos del vector se accederían con `v(end-4:end)`.

Receta. Ejercicio 9

1. Generamos la matriz con la función `magic`.
2. Calculamos la suma de cada una de sus filas.
3. Calculamos la suma de cada una de sus columnas.

4. Calculamos la suma de los elementos de sus diagonales principales: una será la traza de la matriz, y la otra la traza de la matriz resultante de voltear la matriz de izquierda a derecha.

Receta. Ejercicio 10

Para resolver este ejercicio tenemos que pensar en cómo funciona el producto de matrices. Si pensamos en multiplicar una matriz diagonal por una matriz cualquiera, es fácil ver que el resultado es que nosotros esperamos en el ejercicio. Es decir, en la primera fila de la matriz resultante los valores serán los valores de la primera fila de la segunda matriz en el producto, multiplicados por el primer elemento de la diagonal en la matriz diagonal, en la segunda fila pasará lo mismo pero con la segunda fila de la segunda matriz en el producto y el segundo elemento en la diagonal de la matriz diagonal, y así sucesivamente.

Por lo tanto para obtener el resultado deseado no tenemos más que construir una matriz diagonal con los elementos de la primera columna de la matriz original (que son los primeros elementos de cada fila), y multiplicar dicha matriz diagonal por la matriz original.

Receta. Ejercicio 11

1. Obtener el resto y el cociente de la división entera de dividir el saldo por el precio de un lápiz: el cociente será el número máximo de lápices y el resto el dinero sobrante. Con este dinero sobrante se calculan cuántas gomas se pueden comprar, nuevamente calculando el cociente de la división entera.

2. El segundo apartado se hace de forma análoga.

Receta. Ejercicio 12

1. Definir un vector con los números de 0 a 9999, que son los que vamos a necesitar en las sumatorias que vamos a calcular.

2. Calcular 3 vectores con los términos de cada una de las sumatorias. *Observa que no se puede calcular un vector único de términos y luego sumar los 100 primeros o los 1000 primeros, puesto que para cada* n *los términos de la sumatoria cambian ligeramente, ya que en el exponente tenemos* n *dividiendo.*

    - Tener en cuenta que la función `exp` aplicada sobre un vector actúa elemento a elemento, devolviendo un vector en el que cada elemento es el resultado de e^x donde x es cada uno de los elementos del vector.

3. Mostrar para cada n la suma de los términos que hemos calculado.

Receta. Ejercicio 13

1. Declaramos un rango de valores para la x que vaya de 0 a 10π, con saltos lo suficientemente pequeños como para que las gráficas se vean con suficiente resolución.
2. Dibujamos cada una de las grácias.
3. Las líneas rectas son funciones que para cualquier x devuelve el valor de su altura ($\frac{\sqrt{2}}{2}$ y $-\frac{\sqrt{2}}{2}$, según corresponda).

Receta. Ejercicio 14

Este ejercicio no precisa de receta.

Receta. Ejercicio 15

1. Definimos una matriz 3x3 como queramos.
2. Calculamos el determinando usando la formula anterior, en la que tendremos que acceder a los elementos de la matriz segun indica la formula.
3. Usamos la funcion `det` para calcular el determinante con la funcionalidad de Matlab\Octave .

Receta. Ejercicio 16

1. Definimos la matriz y el vector con la información del enunciado.
2. Multiplicamos la matriz por el vector de precios traspuesto. Esto, por como funciona el producto de matrices nos dará un vector columna con lo gastado por cada cliente en total.
3. Calcular la media del producto de la columna de gastos referente al Producto B por el precio del producto B.
4. Suma el vector obtenido en el primer apartado.

Receta. Ejercicio 17

1. Definir las dos matrices con las funciones `zeros` y `ones`.
2. Calcular su matriz inversa con la función `inv`.

Receta. Ejercicio 18

1. Declarar el vector con las alturas.
2. Mostrar el máximo y el mínimo de ese vector.
3. Ordenar el vector de menor a mayor y calcular la media de las 5 primeras posiciones de ese vector ordenado.
4. Ordenar el vector de mayor a menor y calcular la suma de las 3 primeras posiciones.

Receta. Ejercicio 19

1. Obtenemos un vector de 100 números aleatorios en el rango $[0,1]$ con la función.
2. Usamos sobre este vector la fórmula del enunciado donde ahora partirmos del rango $[0,1]$ y pasamos al rango $[5,10]$.
3. Pintamos los valores.

Receta. Ejercicio 20

1. Definir un vector con los salarios.
2. Obtener una permutación aleatoria de los números del 1 al 6.
3. Barajar el vector, usando la permutación anterior para acceder a los elementos del vector por ese orden.
4. Aplicar los cambios en el vector de salario una vez barajado.

Receta. Ejercicio 21

1. Definimos un vector con los valores de la primera fila.
2. Definimos la matriz haciendo uso de las operaciones vectoriales.
3. Dibujamos las gráficas dándole distintos colores a cada una, accediendo para ello a las filas correspondientes de la matriz.

Receta. Ejercicio 22

Este ejercicio no precisa de receta.

Receta. Ejercicio 23

Este ejercicio no precisa receta, salvo destacar la diferencia entre las instrucciones `A.^2` y `A^2`, donde la primera nos da el la matriz resultante de elevar cada elemento de `A` al cuadrado, y la segunda el resultado de multiplicar `A` por ella misma. Siendo la última la que hemos de emplear en este ejercicio.

Receta. Ejercicio 24

Este ejercicio no precisa receta.

Receta. Ejercicio 25

1. Definimos las matrices.
2. Calculamos el precio de cada producto en cada país después de aplicar los factores, calculando el producto elemento a elemento de ambas matrices.
3. Calculamos los 4 puntos que se piden en el ejercicio.

Receta. Ejercicio 26

1. Definimos el vector.
2. Para la primera parte con un bucle `for` vamos recorriendo los elementos del vector, teniendo en cuenta que **necesitamos modificarlos**.
3. Para cada elemento del vector, si es múltiplo de 5, lo cambiamos de signo.
4. Para la segunda parte, accedemos a las posiciones donde hay valores múltiplos de 3, y le asignamos el resultado de multiplicar las mismas posiciones por 2.

Receta. Ejercicio 27

1. La función a diseñar recibe como parámetro la matriz que vamos a analizar. Y en este caso es una función que devuelve **dos valores**.
2. Obtengo una matriz de booleanos donde se indique qué elementos de la matriz son múltiplos de 2 o de 3, pero no de ambos. Y uso esos booleanos (que son ceros y unos) para obtener el número total de elementos que cumplen esa condición, sumando los booleanos (esta suma será la que devuelva como uno de los dos valores devueltos por la función).
3. Para el segundo paso obtengo el número de filas y de columnas de la matriz en dos variables.
4. Con un bucle `for` doble recorro la matriz: el primer bucle recorrerá las filas, y el bucle interior las columnas. Para cada elemento de la matriz (`M (i,j)`), si cumple que es múltiplo de 5 y de 4, aumento en 1 una variable (que es la que devuelvo como segundo valor) que lleva la cuenta de los elementos de la matriz que cumplen esto.

Receta. Ejercicio 28

1. Calculamos el número de valores positivos y negativos en el vector, que lo abremos definido gracias a la función `rand` y a la fórmula del cambio de escala. Para pasar de un valor x en el rango $[a, b]$ a otro en el rango $[c, d]$ empleamos la fórmula:
$$\frac{(d-c)(x-a)}{b-a} + c$$
Además, rendodeamos lo elementos del vector al entero más cercano, para así tener un vector de enteros y no de decimales.

2. Mostramos el número de valores nulos en el vector: esto es algo que hacemos en cualquier caso con lo que va fuera de cualquier estructura condicional.

3. A continuación, siguiendo el enunciado comprobamos qué acciones realizamos según qué caso, y las ponemos en un if else. Tendremos una estructura `if-elseif-else`, como estructura condicional principal para distinguir los 3 casos que pueden darse según el número de elementos positivos y negativos.

4. Dentro de cada caso hacemos lo que se pide, empleando otra estructura condicional cuando sea necesario.

5. Para comprobar el correcto funcionamiento de nuestro código, o del que se da como ejemplo, se pueden definir vectores a manos con unos pocos valores, intentando ver si según el vector que definamos se ejecuta lo que el programa a de ejecutar.

Receta. Ejercicio 29

Para definir una función que calcule con estructuras repetitivas el máximo de un vector:

1. La función recibe el vector como parámetro. Y devuelve un solo valor, que será el máximo del vector.

2. Entonces inicialmente el máximo será el primer elemento del vector.

3. Luego iremos recorriendo el vector. En este caso podemos recorrer directamente los valores sin hacer `v(i)` ya que sólo no nos interesan los valores del vector, y no vamos a modificar el vector en sí.

4. Para cada valor del vector, si es mayor que el máximo actual que hemos encontrado, entonces el máximo pasará a ser ese valor.

La función para calcular el mínimo se definiría de forma análoga. Ahora para el script donde se usan las funciones:

1. Definimos un vector de números positivos, como queramos.

2. Le pedimos al usuario un primer valor.

3. Y mientras que el usuario introduzca valores mayor o iguales que cero:

   a) Si el valor introducido está en el rango [minimo, maximo], lo sumamos a la suma actual de los valores en el rango.

4. Finalmente mostramos la suma calculada cuando el usuario haya introducido un valor negativo, parando así el bucle en el que se le van pidiendo valores.

Receta. Ejercicio 30

1. La función recibe como argumento la matriz a la que le vamos a calcular la media de su triangular superior. Al igual que en Matlab la función `mean` recibe por ejemplo el vector al que le vamos a calcular su media.

2. La función devolverá un valor; la media de la triangular superior de la matriz. Al igual que la función `mean` devuelve la media del vector que le pasemos.

3. Para calcular la media recorremos los elementos de la triangular superior:

   - En la fila i estos elementos serán los que van de la columna i hasta el final.
   - Por tanto el primer bucle recorrerá las filas y el segundo, dentro del primero, recorrerá las columnas que vayan desde la columna i (si estamos en la fila i), hasta el final.

4. Vamos acumulando la suma de estos elementos en una variable, y también llevamos la cuenta de cuántos elementos hemos sumado en otra variable. Con estos dos datos finalmente calculamos la media, que será el valor que devolveremos.

Receta. Ejercicio 31

Para la función que comprueba si un número es primo. Esta función recibirá un número, y devolverá un booleano indicando si el número es primo o no. Entonces:

1. El booleano empezará siendo `true`.

2. Para cada entero mayor que 1 y menor que el número introducido, comprobamos si ese entero divide al número introducido, en cuyo caso el número tendría un divisor distinto del 1 y de él mismo, con lo que el booleano que indica si el número es primo o no pasaría a ser `false`.

El script que se pide no es más que una estructura condicional anidada donde seguiremos las indicaciones que se dan en el enunciado para ver cómo anidamos las estructuras condicionales.

Receta. Ejercicio 32

1. Nuestra función recibirá como parámetro el vector a dibujar. Y en este caso será una función que no devolverá nada, ya que lo único que queremos que haga es dibujar el gráfico de barras correspondiente.

2. En la función lo primero que hacemos es calcular la altura máxima del gráfico que se corresponderá con el valor máximo del vector que pasamos como argumento.

3. Ahora iremos dibujando el gráfico de arriba a abajo, es decir de la fila más alta, correspondiente a la altura máxima del gráfico, a la fila uno.

4. Para cada una de las filas del gráfico, recorremos el vector. Si en una posición tenemos que la altura de esa barra es mayor o igual que la fila actual, entonces dibujamos un *, y si no dibujamos un espacio en blanco.

Receta. Ejercicio 33

1. Se pide el número de jugadores y se genera una cantidadMaxima aleatoria en el [0,1000].

2. Ahora para cada jugador:

   a) Se le indica su cantidad actual. Y se le pregunta si quiere continuar.

   b) Se le seguirá preguntando si quiere continuar mientras que no introduzca una respuesta válida que es "S" o "N".

   c) Si responde "S" se aumentará su cantidad actual un número aleatorio en el intervalo [0, cantidadMaxima / 10], y se vuelve al inicio de este bucle.

   d) Si responde "N" se muestra la cantidad total acumulada por el jugador, y se pasa al siguiente jugador.

Receta. Ejercicio 34

La función recibirá una matriz como argumento, y devolverá la traza de dicha matriz.

La traza de una matriz se puede ver como la suma de aquellos elementos cuya posición tiene la fila y columna igual. Entonces:

1. Calculamos el mínimo entre el número de filas y el de columnas de la matriz, así nos aseguraremos de no tener un fallo en las dimensiones ya que si por ejemplo tenemos una matriz con 2 filas y 3 columnas, el elemento M(3,3) no existe, y tendríamos un fallo en las dimensiones.

2. Teniendo en cuenta ese mínimo hacemos un bucle para sumar los elementos `M(i,i)` de la matriz.

Receta. Ejercicio 35

Esta función es claro que no necesita recibir ningún parámetro, puesto que lo único que hace es generar un entero aleatorio, ya que el entero no depende de un valor o valores que le pasemos a la función.

La función sí que devuelve un valor, un entero aleatorio entre el 0 y el 100, para ello nos podemos ayudar de la función `rand` de Matlab.

Receta. Ejercicio 36

Tenemos una función que recibe un solo parámetro, que es el vector al que le vamos a buscar el máximo, y que devuelve dos valores: el máximo del vector y su posición.

Lo que haremos en la función será recorrer los elementos del vector, e ir almacenando el máximo valor encontrado hasta el momento en la variable que devolverá el máximo, y también se devolverá la posición en la está ese elemento en la variable que devolverá la posición.

En el script que usa esta función simplemente:

1. Obtenemos el máximo y la posición del vector.
2. Mostramos el máximo.
3. Recorremos las posiciones del vector, y para aquellas que no sean la posición del máximo cambiamos el valor por cero.

Receta. Ejercicio 37

En este ejercicio no tenemos más que seguir las indicaciones que se dan en el enunciado. Teniendo en cuenta:

- Que algunos puntos se pueden agrupar bajo la misma condición, y no hay que hacer dos bloques condicionales distintos.
- Que queremos que la entrada del usuario a la pregunta de si quiere o no estudiar un idioma sea una cadena que puede ser solamente "S" o "N", con lo que habrá que forzar la entrada con un `while`.

Receta. Ejercicio 38

1. Creamos la matriz como queramos. Por ejemplo enteros aleatorios en el [0,10].
2. Contamos cuántos elementos cumplen la condición usando `sum` dos veces para sumar los 1 del vector de la matriz de booleanos que obtendremos.
3. Obtenemos el número de filas y de columnas de la matriz.

4. Hacemos un `for` doble para recorrer la matriz, el primer bucle recorrerá las filas y el segundo las columnas. Para cada fila almacenamos el valor del primer elemento de esa fila, que será el valor por el que queremos multiplicar todos los elementos de esa fila (si no, multiplicaríamos ese elemento por si mismo en el segundo bucle, y multiplicaríamos el resto de los elementos de la fila por su cuadrado, y no por el primer elemento original).

5. En el segundo bucle recorremos las columnas de cada fila, y multiplicamos sus elementos por el valor almacenado en la otra variable.

6. Para seleccionar los elementos de la matriz que son múltiplos de 4 simplemente usamos el acceso por condiciones que hemos visto en el tema de estructuras condicionales.

Receta. Ejercicio 39

Tenemos que tener en cuenta que dada una matriz M su matriz traspuesta M' cumple que $M(i,j) = M'(j,i)$, es decir, se intercambian filas por columnas. Dicho esto, suponiendo que vamos a recorrer los elementos de la triangular superior de la matriz, sin incluir la diagonal:

1. Obtenemos el número de filas y columnas de la matriz.

2. Hacemos un bucle `for` doble que recorrerá la matriz.

3. El primer bucle recorrerá las filas.

4. El segundo recorrerá las columnas correspondientes a los elementos de la triangular superior de la matriz sin incluir la diagonal principal. Es decir esas columnas después de la columna perteneciente a la diagonal principal.

5. Ahora intercambiaremos el valor $M(i,j)$ de la matriz con el valor $M(j,i)$ de la matriz.

Receta. Ejercicio 40

La función recibirá dos valores, el dividendo y el divisor. Y devolverá también dos valores, el cociente y el resto de la división entera. Para calcular ese cociente y el resto no tenemos más que implementar dentro de la función el algoritmo que se describe en el enunciado.

Recordemos que para recoger los valores de una función que devuelve varios valores entonces tenemos que hacer algo como `[Q,R] = ej40(10,5)`.

3. Soluciones

Solución. Ejercicio 1

```
radios = [10 5 8 6 20 50];
fprintf('Hay %d circulos en el vector.\n', length(radios))
pi*radios.^2 % Observa el uso del operador de exponenciacion elemento a elemento.
2*pi*radios
```

Solución. Ejercicio 2

```
%{
A continuacion defino la matriz con las distancias.
No es necesario poner cada fila en una nueva linea, lo hago para aumentar la legibilidad del codigo.
%}
distancias = [0 10 20 30 40;
15 0 20 50 60;
20 30 0 80 10;
50 40 20 0 10;
20 20 54 21 0]

fprintf('La suma de las distancias de cada ciudad a ella misma es: %d.\n', trace(distancias))
fprintf('La distnacia maxima en la matriz es: %d.\n', max(max(distancias)))
fprintf('La ruta mas larga desde la ciudad A es de %d km.\n', max(distancias(1,:)))
fprintf('La ruta mas larga hasta la ciudad C es de %d km.\n', max(distancias(:,3)))
```

Solución. Ejercicio 3

```
notas = [5 7 10;
6 4 3;
8 5 8;
9 8 9];

fprintf('La nota del Estudiante2 en Lengua es %f.\n', notas(2,2))

%{
Puedo usar display ya que no necesito darle ningun formato a la cadena a imprimir.
No voy a introducir ningun valor en ciertos huecos que ponga. O voy a usar algun
caracter especial para por ejemplo poner un salto del linea con el \n
%}
display('La desviacion tipica en cada asignatura es:')
%Observa como funciona el fprintf cuando le pasamos un vector para rellenar un hueco.
```

```
%Es como si llamasemos al fprintf una vez por cada uno de los elementos del vector.
fprintf('%f ', std(notas))
fprintf('\n')

display('La media de cada estudiante redondeada al alza es:')
fprintf('%f ', ceil(mean(notas, 2)))
fprintf('\n')

bar(notas)
xlabel('Estudiante')
ylabel('Calificacion')
legend('Matematicas', 'Lengua', 'Filosofia', 'BEST')
title('Notas de los estudiantes')
```

Solución. Ejercicio 4

```
t0 = input('Dame un instante de tiempo en el intervalo [1,3]: ');
y0 = abs(sin(t0^t0) / 2^((t0^t0 - pi/2)/pi));
fprintf('La altura de la pelota en el instante %f es %f.\n', t0, y0)

t = 1:0.01:3;
y = abs(sin(t.^t) ./ 2.^((t.^t - pi/2)/pi));
plot(t,y)
xlabel('Tiempo')
ylabel('Altura')
title('Evolucion de la altura en el tiempo')
```

Solución. Ejercicio 5

```
gastos = rand(5,12) * 1000;
ingresos = rand(5,12) * 1000;

neto_mensual = ingresos - gastos;
display('El neto mensual de cada persona es:')
neto_mensual

display('El neto anual de cada persona es:')
sum(neto_mensual, 2)

display('El neto maximo mensual de cada persona es:')
max(neto_mensual, 2)

display('El neto minimo mensual de cada persona es:')
min(neto_mensual, 2)
```

```
fprintf('En media las personas han gastado en este anio %f.\n', mean(sum(gastos, 2)))

display('La razon ingresos-gastos mensual de cada persona es:')
ingresos ./ gastos
```

Solución. Ejercicio 6

```
nombre = input('Dime tu nombre: ', 's');
notas = [5 6 8 4 3 8 7];

fprintf('%s, la media de tus calificaciones es de %f.\n', nombre, mean(notas))
fprintf('%s, tu mejor nota es %f.\n', nombre, max(notas))
fprintf('%s, tu peor nota es de %f.\n', nombre, min(notas))

fprintf('%s, aqui puedes ver un grafico con la evolucion de tus notas.\n', nombre)
plot(notas)
```

Solución. Ejercicio 7

```
n = 1:1000;
terminos = (4*n.^2) ./ (4*n.^2 - 1); %Observa el uso de los operadores elemento a elemento.
productorios = 2*cumprod(terminos);

display('Los resultados de productorios parciales son:')
productorios

plot(productorios)
```

Solución. Ejercicio 8

```
v = [-1 5 6 8 9 10 20 -5 6 8];

v(end:-1:1)
sort(v)

v(1:2:end) = abs(v(1:2:end));
display('Tras poner en valor absoluto los elementos en las posiciones impares:')
v

v(3:3:end) = 2 * v(3:3:end)
display('Tras duplicar lo elementos en las posiciones multiplo de 3:')
v

```

```
fprintf('La suma de los 5 primeros elementos es %d.\n', sum(v(1:5)))

fprintf('El productor de los elementos en las posiciones pares es %d.\n', prod(v(2:2:end)))

v(end-4:end) = v(end-4:end) - 1;
display('Tras restar 1 a los 5 ultimos elementos:')
v
```

Solución. Ejercicio 9

```
M = magic(5) %No pongo ; porque quiero ver la matriz generada.

display('La suma de cada columna:')
sum(M)

display('La suma de cada fila:')
sum(M, 2)

fprintf('La suma de una diagonal es %d.\n', trace(M))
fprintf('La suma de la otra diagonal es %d.\n', trace(M(:,end:-1:1)))
```

Solución. Ejercicio 10

```
M = [1 2 3;
4 5 6;
7 8 9];

diag(M(:,1)) * M %Otra alternativa es M(:,1) .* M.
```

Solución. Ejercicio 11

```
precio_lapiz = 1.5;
precio_goma = 2;

nombre = input('Dime tu nombre: ', 's');
saldo = input('Dime el saldo del que dispones: ');

lapiz_max = floor(saldo / precio_lapiz); %Otra opcion es usar la funcion fix: fix(saldo / precio_lapiz), que nos da el resultado de la division entera, es decir, sin decimales.
gomas_con_sobrante = floor(rem(saldo, precio_lapiz) / precio_goma);

fprintf('%s, puedes adquirir un maximo de %d lapices. Y con el resto puedes comprar %d gomas.\n', nombre, lapiz_max, gomas_con_sobrante)
```

```

gomas_max = floor(saldo / precio_goma);
lapices_con_sobrante = floor(rem(saldo, precio_goma) / precio_lapiz);

fprintf('%s, puedes adquirir un maximo de %d gomas. Y con el resto puedes comprar %d lapices.\n', nombre, gomas_max, lapices_con_sobrante)
```

Solución. Ejercicio 12

```
n = 0:9999;
terms100 = exp(2*pi*i*n(1:100)/100);
terms1000 = exp(2*pi*i*n(1:1000)/1000);
terms10000 = exp(2*pi*i*n(1:10000)/10000);

fprintf('Para n = 100 el resultado es: %f.\n', sum(terms100));
fprintf('Para n = 1000 el resultado es: %f.\n', sum(terms1000));
fprintf('Para n = 10000 el resultado es: %f.\n', sum(terms10000));
```

Solución. Ejercicio 13

```
x = 0:0.1:10*pi;

plot(x, cos(x), 'g-')
%{
Hace que no se resetee el panel de dibujo al usar nuevamente plot.
Por lo que lo que pintemos con plot se dibujara encima de lo que ya hay.
%}
hold on
plot(x, sin(x), 'b*')
%{
No es necesario emplear un vector con el valor raiz(2)/2 tantas veces como
elementos tengas el valor de la x, basta con ponerlo una vez.
%}
plot(x, sqrt(2)/2, 'r')
plot(x, -sqrt(2)/2, 'r')

legend('cos', 'sin')

%Otra alternativa sin usar hold on es usar un solo plot.
%plot(x, cos(x), 'g-', x, sin(x), 'b*', x, sqrt(2)/2, 'r', x, -sqrt(2)/2, 'r')
```

Solución. Ejercicio 14

```
sqrt(2^10) + exp(10/5+4) * log(5)
log10(100) * (nthroot(128,4) + 50)
sin(log(20)) * csc(pi / 2)
(asin(1) - cot(3*pi/2) + cos(pi/8)) + abs(5-8) * log(30)
```

Solución. Ejercicio 15

```
A = rand(3,3);

%{
Observa como se puede partir una linea larga en varias lineas usando ... al final
de la linea. De este modo una linea de codigo aparece en varias lineas, pero funciona
como si estuviese escrita en una sola.
%}
detConFormula = A(1,1)*(A(2,2)*A(3,3) - ...
A(2,3)*A(3,2)) - A(1,2)*(A(2,1)*A(3,3) - ...
A(2,3)*A(3,1)) + A(1,3)*(A(2,1)*A(3,2) - A(2,2)*A(3,1))

fprintf('El determinante con la formula es: %f.\n', detConFormula)
fprintf('El determinante con la funcion det es: %f.\n', det(A))
```

Solución. Ejercicio 16

```
gastos = [100 200 50 40;
50 80 10 5;
50 50 60 210];
precios = [50 20 10 3];

gastoPorCliente = gastos * precios';

display('Lo gastado por cada cliente en total es:')
gastoPorCliente

fprintf('En media los clientes han gastado %f en el Producto B.\n', mean(gastos(:,2) * precios(2)))
fprintf('Lo gastado en total por todos los clientes es: %f.\n', sum(gastoPorCliente))
```

Solución. Ejercicio 17

```
ceros = zeros(10);
unos = ones(5);

inv(ceros)
inv(unos)
```

Solución. Ejercicio 18

```
alturas = [1.5 1.9 1.35 2 1.6 1.72 1.7 1.85 1.4 1.45];

fprintf('La persona mas baja mide %f m.\n', min(alturas))
fprintf('La persona mas alta mide %f m.\n', max(alturas))
fprintf('La media en altura de las 5 personas mas bajas es de %f m.\n',...
 mean(sort(alturas)(1:5)))
fprintf('La suma de las alturas de las 3 personas mas altas es de %f m.\n',...
 sum(sort(alturas, 'descend')(1:3))) %Observa como se ordena un vector de mayor a menor.
```

Solución. Ejercicio 19

```
values = rand(1,100);
values = 5 + 5*values;
plot(values)
```

Solución. Ejercicio 20

```
salarios = [1100 2500 1350 2000 2000 1800];
permutacion = randperm(6);
salarios = salarios(permutacion);

salarios(2:2:end) = 1.10 * salarios(2:2:end);
salarios(1:3) = salarios(1:3) + 100;
salarios(end) = 2 * salarios(end);
```

Solución. Ejercicio 21

```
valores = 1:0.2:10;
M =  [valores; sqrt(valores); sin(valores).*cos(valores); log(valores)];

plot(M(1,:), M(2,:), 'g')
hold on
plot(M(1,:), M(3,:), 'r')
plot(M(1,:), M(4,:), 'b')
legend('sqrt', 'sin*cos', 'log')
```

Solución. Ejercicio 22

```
M = rand(5,10);

fprintf('La suma de los elementos en las filas pares y columnas multiplo de 3 es: %f.\n',...
sum(sum(M(2:2:end, 3:3:end))))
```

```
fprintf('La suma de los elementos en las filas 1, 2 y 5, que estan en columnas pares es: %f.\n',...
sum(sum(M([1 2 5], 2:2:end))))
matrizCubos = M.^3;
```

Solución. Ejercicio 23

```
A = [2 3 -5; 10 1 0; 5 5 6];
B = [4 0 1; -9 -2 5; 0 1 7];

A*B
B*A
A.*B
A^2*B
A*inv(A)
A+B
det(A) * B
A / trace(B)
```

Solución. Ejercicio 24

```
uniforme = rand(1,1000);
normal = randn(1,1000);

fprintf('La media del vector generado con rand es %f.\n', mean(uniforme));
fprintf('La media del vector generado con randn es %f.\n', mean(normal));
```

Solución. Ejercicio 25

```
preciosBase = [100 200 300; 400 300 200; 500 400 100];
factores = [1.1 0.5 0.3; 0.5 2 1.2; 1 1 1.5];

preciosFinales = preciosBase .* factores;

fprintf('Precio medio del Produco A despues de aplicar los factores: %f.\n',
mean(preciosFinales(:,1)))

fprintf('Diferencia media absoluta en el precio del Producto B antes y despues de aplicar los factores: %f.\n',
mean(abs(preciosFinales(:,2) - preciosBase(:,2))))

fprintf('Precio medio de los productos en el Pais 2 despues de aplicar los factores: %f.\n',
mean(preciosFinales(2,:)))

fprintf('Diferencia media absoluta de precios entre los paises 1 y 3 despues de aplicar los factores: %f.\n',
mean(abs(preciosFinales(1,:) - preciosFinales(3, :))))
```

Solución. Ejercicio 26

```
v = 1:100;

for i = 1:length(v)
  if (rem(v(i), 5) == 0)
        v(i) = -v(i);
  end
end

v(rem(v, 3) == 0) = 2 * v(rem(v,3) == 0);

display(v)
```

Solución. Ejercicio 27

```
function [n1, n2] = ej27(M)
  n1 = sum(sum(xor(rem(M, 2) == 0, rem(M, 3) == 0))); %
      Observad que hay que usar sum dos veces, y entended por
       que.

  n2 = 0;

  [nfilas, ncols] = size(M); %Obtengo el numero de filas y
      columnas de la matriz.

  for i = 1:nfilas
        for j = 1:ncols
          if (rem(M(i,j), 5) == 0 && rem(M(i,j), 4) == 0)
                n2 = n2 + 1;
          end
        end
  end
end
```

Solución. Ejercicio 28

```
v = rand(1, 100);
v = round(-100 + 200*v);
display(v)
%{
Declaramos las siguientes variables para no tener que
    calcular el numero de
positivos y negativos varias veces, haciendo el codigo mas
    legible.
%}
nPositivos = sum(v >= 0);
nNegativos = sum(v < 0);

```

```
% Mostramos la siguiente info. antes de modificar el vector posiblemente en el else.
fprintf('El numero de valores nulos en el vector es %d.\n', sum(v == 0))

if nPositivos > nNegativos
  display('Hay mas valores positivos que negativos.')

  if sum(rem(v,2) ~= 0) < sum(rem(v,2) == 0)
        display('Hay mas valores pares que impares.')
  else
        display('Hay mas valores impares que pares.')
  end
elseif nPositivos < nNegativos
  display('Hay mas valores negativos que positivos.')
  fprintf('Hay %d multiplos de 3 en el vector.\n', sum(rem(v,3) == 0))

  if  sum(abs(v(rem(v,5) == 0))) > sum(v(v > 0))
        display('La suma de multiplos de 5 en valores absoluto, es mayor que la suma de elementos positivos')
  end
else
  display('Hay el mismo numero de valores positivos que negativos.')
  v(v < 0) = 0;
  v(v > 0) = -v(v > 0);
end
```

Solución. Ejercicio 29

Archivo maximum.m

```
function [maximo] = maximum(v)
  maximo = v(1);

  for elem = v
        if maximo < elem
          maximo = elem;
        end
  end
end
```

Archivo minimum.m

```
function [minimo] = minimum(v)
  minimo = v(1);

  for elem = v
        if minimo > elem
```

```
            minimo = elem;
        end
    end
end
```

El script donde se usan ambas funciones.

```
v = [1 2 5 15 2 6 5 8 10];
suma = 0;

minimo = minimum(v);
maximo = maximum(v);

valor = input('Introduce un valor:');

while valor >= 0
  if minimo <= valor && valor <= maximo
        suma = suma + valor;
  end
  valor = input('Introduce un valor:');
end

fprintf('La suma de los elementos introducidos por el
    usuario dentro de rango es %f.\n', suma)
```

Solución. Ejercicio 30

```
function [media] = ej30(M)
  %{
  Como sabemos que es cuadrada nos basta con esto
  para obtener el numero de filas y de columnas.
  %}
  n = length(M);
  nelementos = 0;
  suma = 0;

  for i = 1:n
        for j = i:n
          suma = suma + M(i,j);
          nelementos = nelementos + 1;
        end
  end

  media = suma / nelementos;
end
```

Solución. Ejercicio 31

La función `esPrimo` que ha de guardarse en un archivo `esPrimo.m`.

```
function [es] = esPrimo(n)
```

```
  es = true;

  for div = 2:(n-1)
        if rem(n, div) == 0
          es = false;
        end
  end
end
```

El script donde se usa la función `esPrimo`.

```
num = input('Introduce un numero:');

if rem(num, 2) == 0 && num > 10
  display('El numero es par y mayor que 10.')

  if rem(num, 5) == 0
        display('El numero es multiplo de 5.')
  else
        display('El numero no es multiplo de 5.')
        if esPrimo(num)
          display('El numero es primo.')
        else
          display('El numero no es primo.')
        end
  end
else
  display('El numero no es par o no es mayor que 10.')
end
```

Solución. Ejercicio 32

```
function ej32(v)
  altura = max(v);

  for i = altura:-1:1
        for elem = v
          if i <= elem
                fprintf('*')
          else
                fprintf(' ')
          end
        end
        fprintf('\n')
  end
end
```

Solución. Ejercicio 33

```
nJugadores = input('Indique el numero de jugadores:');
```

```
cantidadMaxima = rand(1,1) * 1000;
fprintf('La cantidad maxima es: %f.\n', cantidadMaxima)

for jugador = 1:nJugadores
  cantidadJugador = 0;
  fprintf('Su cantidad actual es %f.\n', cantidadJugador);

  opcion = input('Desea continuar? S/N: ', 's');
  while opcion ~= 'S' && opcion ~= 'N'
        opcion = input('Desea continuar? S/N: ', 's');
  end

  while opcion == 'S'
        cantidadJugador = cantidadJugador + rand(1,1) * (cantidadMaxima / 10);

        fprintf('Su cantidad actual es %f.\n', cantidadJugador);

        opcion = input('Desea continuar? S/N: ', 's');
        while opcion ~= 'S' && opcion ~= 'N'
          opcion = input('Desea continuar? S/N: ', 's');
        end
  end

  fprintf('La cantidad final del Jugador %d es: %f.\n', jugador, cantidadJugador);
end
```

Solución. Ejercicio 34

```
function [traza] = ej34(M)
  traza = 0;
  n = min(size(M));

  for i = 1:n
        traza = traza + M(i,i);
  end
end
```

Solución. Ejercicio 35

```
function [numero] = ej35
  numero = round(rand(1,1) * 100);
end
```

Solución. Ejercicio 36

Listing 1: El archivo maxPos.m

```
function [maximo, pos] = maxPos(v)
```

```
  maximo = v(1);
  pos = 1;

  %{
  Observemos que aqui empleamos la posicion del vector para acceder al vector.
  Solo necesitamos los valores del vector, pero dado a que vamos a querer
  obtener la posicion donde se encuentra el maximo, usamos el acceso por indice.
  %}
  for i = 1:length(v)
        if maximo < v(i)
          maximo = v(i);
          pos = i;
        end
  end
end
```

Listing 2: El script que usa esta función.

```
v = [1 5 4 20 -5 6 9 10];
[m, p] = maxPos(v);

fprintf('El maximo del vector es %d.\n', m)

for i = 1:length(v)
  if i ~= p
        v(i) = 0;
  end
end

v
```

Solución. Ejercicio 37

```
nombre = input('Introduzca su nombre: ', 's');
edad = input('Introduzca su edad: ');
peso = input('Introduzca su peso: ');

if edad < 18
  altura = input('Introduzca su altura en metros: ');
  fprintf('%s, tu IMC es %f.\n', nombre, peso / altura^2)

  estudiar = input('Desea estudiar un idioma? S o N: ', 's');
  while estudiar ~= 'S' && estudiar ~= 'N'
        estudiar = input('Desea estudiar un idioma? S o N: ', 's');
  end
```

```

  if estudiar == 'S'
        fprintf('%s deberias estudiar sueco.\n', nombre)
  else
        fprintf('%s deberias elegir musica como actividad extraescolar.\n', nombre)
  end
elseif 30 <= edad && edad <=50
  salario = input('Introduzca su salario: ');

  if salario < 700
        fprintf('%s deberias alquilar un piso.\n', nombre)
  elseif 700 <= salario && salario <= 1500
        fprintf('%s deberias comprar un piso.\n', nombre)
  else
        fprintf('%s deberias comprar una casa.\n', nombre)
  end
else
  fprintf('%s deberias contratar un plan de pensiones.\n', nombre)

  if peso > 100
        fprintf('%s deberias hacer deporte.\n', nombre)
  end
end
```

Solución. Ejercicio 38

```
M = round(rand(10,10)*10) %matriz de enteros entre 0 y 10.

nelem = sum(sum(rem(M, 2) == 0 | xor(rem(M,3) == 0, rem(M, 5) == 0)));

fprintf('Hay %d elementos que cumplen la condicion.\n', nelem)

[nrows, ncols] = size(M);

for i = 1:nrows
  factor = M(i,1);
  for j = 1:ncols
        M(i,j) = factor * M(i,j);
  end
end

M

M(rem(M,4) == 0)' %lo transponemos para que salta en una fila
```

Solución. Ejercicio 39

```
M = round(rand(3,3)*10)

[nrows, ncols] = size(M);

for i = 1:nrows
  for j = (i+1):ncols
        aux = M(i,j);
        M(i,j) = M(j,i);
        M(j,i) = aux;
  end
end

M
```

Solución. Ejercicio 40

```
function [Q,R] = ej40(dividendo, divisor)
  Q = 0;
  R = dividendo;

  while R >= divisor
        Q = Q + 1;
        R = R - divisor;
  end
end
```