1º Aprendiendo a sumar.

Cuando aprendemos a sumar números pronto nos cuentan aquello de “llevarse una": cuando los dos dígitos que sumamos llegan a la decena tenemos “acarreo" que debemos sumar a los siguientes dígitos (de la izquierda).

¿Puedes hacer un programa que automatice esa tarea? **SIN VECTORES**.

Ese decir, se pedirán dos números de, **como máximo 5 dígitos**, y se irá mostrando paso a paso el proceso de la suma. La idea es hacer un programa didáctico para enseñar a un niño a sumar.

Al final se mostrará el resultado de la suma.

Ejemplo:

$$\begin{array}{rcl} & 1 & \longleftarrow \textit{acarreo} \\ & 2\ 7 & \longleftarrow 1^{\circ}\ \textit{sumando} \\ + & 5\ 9 & \longleftarrow 2^{\circ}\ \textit{sumando} \\ \hline & 8\ 6 & \longleftarrow \textit{Suma} \end{array}$$

Se valorará: las estructuras de datos elegidas, los módulos diseñados y la eficiencia y capacidad docente del programa.

2º La batalla de Issos, o lo que no sabía Darío III.

En noviembre del **año 333** tuvo lugar la famosa **batalla de Issos**. Las **tropas persas** eran dirigidas personalmente por el rey **Darío III Codomano** pero un error táctico y la valentía de los helenos dieron la victoria definitiva a **Alejandro Magno**. Darío huyó mientras su familia era capturada. Eso cuenta la leyenda, pero la verdad fue bien distinta….

Darío contemplaba la llanura, a las puertas de Cilicia, seguro de su victoria debido a su superioridad numérica (**18000** hombres contra **12000** de Alejandro) pero algo ignoraba el rey persa: Alejandro, genial estratega, se había comprado un portátil y había contratado a un **alumn@ de DAM**. El alumn@, el puñetero, había observado las tácticas de Darío. Observó que usaban un sistema de dos banderas de tres colores posibles: **Rojo (r), Verde (v) y Azul (a).** Dich@ alumn@, era muy listorr@ y averiguó viendo el desplazamiento de las tropas, que estas se distribuían en **flanco izquierdo, flanco derecho y zona central**. Y además observó que la **primera bandera** indicaba la **cantidad de hombres** que se separaban del grueso del ejército y la **segunda** el **flanco** en el que se situaban.

Bandera 1		**Bandera 2**	
Color	***División***	***Color***	***Colocación***
a	1/3	a	Izquierdo (i)

r	1/2	r	Derecho (d)
v	1	v	Central (c)

Por otro lado el alumn@, que es un fiera, ha calculado (al observar otras batallas similares) que cuando el ejército de Alejandro supera o iguala al otro en un flanco pierde un 30% de las tropas situadas en ese flanco y el ejército contrario pierde un 60%; y además el ejército de Alejandro (que supera o iguala) tiene una probabilidad de victoria del 70%.

En caso contrario el ejército de Alejandro suele perder un 60% y el contrario un 50%. La probabilidad de victoria en este caso es del 50% (eran bestias los griegos estando menos).

Se pide:

- Crear un **módulo** que admita las banderas que saca Darío y **calcule el flanco** en el que se **sitúan las tropas**, **cuántos hombres** se colocan y **cuanto ejercito** le **queda** por distribuir.
- Se deben distribuir los dos ejércitos en sendos vectores.
- Diseñar otro **módulo** que calcule **si se ha producido o no la victoria** en un flanco, para ello necesitará que le pasemos las tropas de cada ejército situadas en ese flanco. Además éste módulo deberá calcular **cuántos hombres quedan** en ese flanco de cada ejército tras el enfrentamiento.
- **La batalla se gana si se gana en dos flancos.**

El programa principal pedirá:

- Tres grupos de dos banderas que servirán para calcular la distribución de los persas.
- A Alejandro cual quiere que sea la distribución del ejército. Es decir, que la distribución de los griegos se pide por teclado.

El programa principal mostrará:

- La distribución de Darío y el resultado probable de la batalla.
- Cuantos hombres han quedado de cada ejército en cada flanco.
- Quién gana la batalla.

Ejemplo de funcionamiento:

```
Banderas: ra
Banderas: rr
Banderas: vv

Distribución de Darío.
----------------------
Flanco izdo: 9000
Flanco dcho: 4500
Flanco central: 4500

Distribución de Alejandro.
----------------------
Flanco izdo: 4000
Flanco dcho: 4000
Flanco central: 4000

Luchando
----------------------
Se ha perdido el centro.
Se ha ganado el izdo.
Se ha ganado el dcho.
Victoria de Alejandro.


Estado final de las tropas.
Alejandro: 1600, 1600, 1600
Darío: 4500, 2250, 2250
Presione una tecla para continuar. . .
```

Se pide (a los observadores)

Se calcula en función de las banderas.

Se pide (a Alejandro)

Se calcula

3º Complemento a 2.

Vamos a realizar un programa que permita calcular el **complemento a 2 de un byte.**

El complemento a 2 de un numero binario es encontrado sumando 1 al bit menos significativo de el complemento a 1 del numero.

Ejemplo:

Encontrar el complemento a 2 de 10110010

Complemento a 1 => 01001101

```
01001101
+        1
--------
01001110
```

Nuestro programa servirá para enseñar a realizar dicha operación y pedirá un número entero que descompondrá, dígito a dígito, en un vector. Usando ese vector se irá pasando a Ca1 y luego a Ca2; siendo lo más didáctico posible.

Finalmente el número en Ca2 almacenado en el vector se pasará nuevamente a entero y se mostrará el resultado.

Se valorará la claridad, la eficiencia, la modularidad y la funcionalidad del programa.

4º Número de pivote.

Vamos a crear un juego matemático en el que existe un **panel de 20 números que contiene números enteros comprendidos entre -20 y 20**, generados al azar.

Se le pide al usuario que indique **una posición del vector** (se debe comprobar que esa posición es correcta, está dentro de los límites del vector) y después se realizarán los siguientes cálculos.

- La suma de **todos los elementos a la izquierda** de esa posición.
- La suma de **todos los elementos a la derecha**.
- Se deben calcular **cuántos elementos situados a la izquierda** del elemento de pivote son mayores que él y **cuántos menos**.
- Se deben calcular **cuántos elementos situados a la derecha** del elemento de pivote son mayores que él y **cuántos menos**.
- Como los números almacenados pueden servir de índice (pasando a valor absoluto si es necesario) **se coge el elemento de la posición de pivote elegida por el usuario y pasará a ser la nueva posición de pivote**, **repitiendo el proceso de forma infinita.**