```
void vectorAmigos (int M, int v[]) {
/* pre: v es un vector de enteros, de tamaño M */
   int suma,i,j;

   for (i=0; i<N; i=i+1) {
      /* Se calcula en suma el valor */
      /* de la suma de divisores de i */
      for (j=1; j<(i/2); j=j+1) {
         if (i%j==0) {
            suma=suma+i;
         }
      }
      /* El valor almacenado en suma es */
      /* el unico candidato a amigo de i */
      /* Si lo es, se guarda en v[i] y, */
      /* si no, se guarda un cero */

      v[i]=(amigos(i,suma))?0:suma;
   }
/* post: v es un vector en el que v[i] es amigo */
/* de i, si i tiene amigos, 0 en caso contrario */
}
```

14. Dado el siguiente bucle, que trabaja con un vector de `N` reales `v` y un valor real `x`,

```
  ....
  boole iguales=cierto;
  int i=0;

  while ((i<N) && iguales)
    if (v[i]==x)
        iguales=falso;
    else
        i=i+1;
  ....
```

¿es un invariante del bucle el predicado

"`v[k]=x`, para todos los valores de k tales que $0 <= k < i$"?

15. Dado un vector A de caracteres, escribir un algoritmo que indique si la frase almacenada en dicho vector es o no capicúa.

Nota: Se considera que los vectores de caracteres poseen un centinela ('\0') que indica el final de los caracteres válidos del vector.

16. ¿Verdadero o falso? Hay que justificar las respuestas.

a) Las siguientes secuencias hacen exactamente lo mismo.

```
/*Secuencia num. 1*/
i=0;
enc=falso;
while((i<n)&&!(enc)){
  enc=(v[i]==x);
  i=i+1;
}
```

```
/*Secuencia num. 2*/
i=0;
enc=falso;
while((i<n){
  if(!(enc)){
    enc=(v[i]==x);
  }
  i=i+1;
}
```

b) El alumno que escribió la secuencia número 2, obtuvo mejor nota en ese problema que el alumno que escribió la secuencia número 1.

c) La siguiente versión es mucho mejor que las anteriores.

```
/*Secuencia num. 3*/
i=0;
enc=falso;
while((i<n){
  if(!(enc)){
    enc=(v[i]==x);
    i=i+1;
  }
}
```

17. Dado un vector `A` de tipo base carácter y dados dos caracteres `car1` y `car2`, escribir un algoritmo que busque las ocurrencias de `car1` en `A` y las sustituya por `car2`.

18. Dado un vector `A` de tipo base carácter y dado un carácter `c`, escribir un algoritmo que busque las ocurrencias de `c` en `A` y las elimine.

19. Necesitamos una función o procedimiento (**justificar la elección**) para corregir las mayúsculas de un párrafo, de modo que tanto la letra inicial como todas las primeras letras que aparezcan después de un punto estén en mayúsculas (independientemente de los espacios en blanco que pueda haber entre el punto y la letra). Por ejemplo, la corrección de la siguiente cadena

    ``El perro ladra. mi madre canta... quisiera verla. Oigo. no veo nada"

    produciría

    ``El perro ladra. Mi madre canta... Quisiera verla. Oigo. No veo nada"

    a) Implementa una función o procedimiento auxiliar `pasarAMayusculas`. Si se le pasa una letra minúscula, debe devolver su conversión a mayúscula. En otro caso, debe devolver el carácter sin modificar. Recuerda que puedes pasar un carácter `c` de minúscula a mayúscula con la expresión `c-32`.

    b) Utilizando `pasarAMayusculas`, implementa la función o procedimiento `corrigeMayusculas`, que obtenga una versión corregida de una cadena dada.

    c) Escribir un programa que use `corrigeMayusculas` para ir corrigiendo cadenas que se irán leyendo de teclado, finalizando el proceso cuando el usuario teclee una cadena vacía.

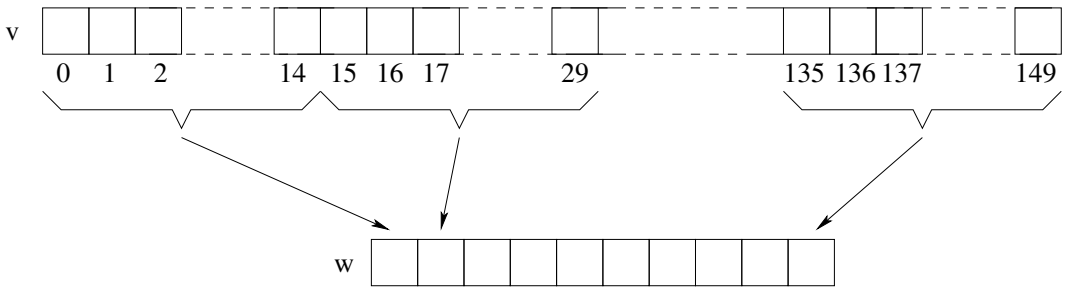
    No olvidéis indicar precondiciones y postcondiciones en todos los apartados.

20. Uno de los métodos más simples para comprimir archivos de música consiste en ir calculando la media aritmética de R valores consecutivos y almacenar el resultado obtenido en otro fichero, también de forma consecutiva. Así, se consigue dividir el tamaño del fichero original entre R. La idea del problema que os proponemos es similar, pero con vectores.

Se dispone de un vector `v` de `N` números reales, siendo `N` múltiplo de `R`, ambas constantes. Y se quiere obtener otro vector `w` de números reales, de tamaño `M=N/R`. Por ejemplo, si `R`=15 y `N`=150, entonces `M` sería 10.

```
#define R 15    /* Reducción del tamaño */
#define N 150   /* Tamaño del vector v, múltiplo de R */
#define M 10    /* Tamaño del vector w, N/R */
```

Se pide escribir una función o procedimiento (hay que justificar la elección) que, dado el vector `v`, obtenga el vector `w`, de modo que cada elemento de `w` sea la media de 15 elementos consecutivos de `v`, tal y como muestra el siguiente esquema:



21. Se sabe que todo número par puede escribirse como la suma de dos números primos. Escribir un algoritmo que, dado un número par, devuelva dos números primos tales que su suma sea el número dado. Indicación: almacenad en un vector los números primos existentes entre el 1 y el número a descomponer.

22. Dados los tipos (N es una constante previamente definida)

    ```
    typedef enum {FALSO, VERDADERO} boole;

    typedef boole vecBoole[N];
    ```

    escribir un algoritmo al que se le pasará un valor de tipo entero (positivo)en el parámetro `a`, y devuelva en un vector `b`, de tipo `vecBoole` el valor de `a` codificado en binario, de forma que asimilaremos el valor `VERDADERO` al dígito binario 1 y el valor `FALSO` al dígito binario 0.

    Por ejemplo, si el valor de a es 35, sabemos que su representación binaria es el número $100011_{(2)}$. Por lo tanto, en b se debería devolver,

    b[0] = `VERDADERO`, b[1] = `VERDADERO`, b[2] = `FALSO`, b[3] = `FALSO`, b[4] = `FALSO`, b[5] = `VERDADERO`, ...

    ¿Debe cumplir `a` alguna condición especial?

23. Dados los tipos

    ```
    typedef enum {FALSO, VERDADERO} boole;

    typedef boole vecBoole[N];
    ```

    donde N es una constante previamente definida, escribir un algoritmo que permita realizar sumas de dos números binarios con acarreo.

    Para ello supondremos que nos dan los números como dos vectores del tipo descrito en el problema anterior. Cada bit se almacenará en una posición del vector, sabiendo que el valor boolea `VERDADERO` se corresponde con el 1 y que el valor boole `FALSO`, con el 0. El resultado será otro número binario, por supuesto.

24. Escribir una función o procedimiento (**justificar la elección**) que, dada una cadena que expresa un valor en hexadecimal empezando con los caracteres `"0x"`, devuelva el correspondiente valor entero.

    Por ejemplo, dada la cadena `"0x123"` debería devolver el número `291`, y dada la cadena `"0x11af3"` debería devolver el número 72435.

    Se supone que la cadena sólo contiene dígitos (`'0'`, `'1'`, ..., `'9'`) y las letras que son dígitos hexadecimales **en minúsculas** (`'a'`, `'b'`, ..., `'f'`), y un `"0x"` al principio.

    Se debe indicar cuál es la precondición y cuál la postcondición.

25. Dada la definición de tipo

    ```
    typedef cualquierTipoBase vector[N];
    ```

    y dado, x, de tipo vector, y que está ordenado, escribir:

    a) Un algoritmo que permita determinar el número de secuencias que hay en x. Se entiende como secuencia cualquier tramo de vector con valores iguales. Así, si por ejemplo tuviéramos el vector

    x = (0, 0, 1, 2, 2, 2, 3, 3, 4, 5, 6, 7, 7, 7, 7, 7, 7, 8, 8, 9, 23)

    el número de secuencias es 11; o si tuviéramos,

    x = (A, A, A, A, G, H, J, J, J, M, M, M, P, P, Z, Z, Z, Z, Z, Z)

    el número de secuencias es 7.

    b) Un algoritmo que permita determinar la secuencia más larga en x, indicando para ello su longitud y la posición en la que comienza. En los ejemplos anteriores, en el primer caso tenemos que la secuencia más larga es la de los 7s, que es de longitud 6 y comienza en la posición 12, y en el segundo es la de las Zs, que es de longitud 6 y comienza en la posición 15.

26. Escribir un algoritmo que indique si un vector tiene dos elementos iguales.

27. Escribir un algoritmo que indique cuántos elementos iguales a un valor dado hay en un vector.

28. Dados dos vectores A y B, de tipo base CARACTER, de tamaño N y M respectivamente (N>M), escribir un algoritmo que busque la primera ocurrencia de B en A; es decir, hay que comprobar si B está contenido en A (los elementos de B son los mismos y están en el mismo orden que un subconjunto de elementos de A) e indicar la posición en la que aparece.

29. Se dispone de un vector con la altimetría de una etapa de una carrera ciclista. El vector contiene la altura en metros en cada punto kilométrico: el índice del vector indica el hito kilométrico y el contenido del vector ofrece la altitud del lugar. Por ejemplo:

Índice:	0	1	2	3	...
Contenido:	500	501	200	300	...

    Los índices del vector van desde 0 hasta el número de kilómetros de la etapa.

    a) Escribir un algoritmo que calcule los kilómetros de subida y los de bajada de la etapa. En el ejemplo, en la etapa hay 2 kilómetros de subida y uno de bajada.

    b) Escribir un algoritmo que calcule el desnivel total de subida que deben superar los ciclistas durante la etapa. Si en un kilómetro se sube menos de 10 metros, este desnivel no debe ser considerado e incluido en los cálculos. En el ejemplo anterior, el desnivel es de 100 metros.

    c) Escribir un programa que calcule el puerto más largo y su desnivel medio (número de metros de desnivel del puerto * 100 / longitud en metros del puerto). Se define un puerto como una subida continua que no contenga bajadas ni tramos llanos.

30. Escribir un algoritmo que, dados los coeficientes de una ecuación de segundo grado, $ax^2 + bx + c = 0$, permita obtener sus raíces, sean estas reales o imaginarias.

31. Para almacenar los datos de cada participante en una competición de lanzamiento de disco, se ha diseñado la siguiente estructura de datos:

```
#define TAM1 40
#define TAM2 20

typedef struct {
   char nombre[TAM1];
   char pais[TAM2];
   int dorsal;
   float lanza1, lanza2, lanza3;
} tLanzador;
```

    a) escribir una función que, dado un lanzador, calcule la media de sus 3 lanzamientos.

    b) escribir una función que, dado un lanzador, devuelva su lanzamiento más largo.

    c) escribir un procedimiento que lea por teclado los datos de un lanzador y los devuelva en un parámetro de salida de tipo `tLanzador`.

    d) suponiendo que en la competición hay inscritos 50 lanzadores y dada la siguiente definición de tipo,

    ```
    #define N 50

    typedef tLanzador tCompeticion[N];
    ```

    hay que escribir una función que devuelva el índice del participante con el mejor lanzamiento.

    e) con la misma definición de `tCompeticion`, escribir una función que devuelva el índice del participante que llevaría el "premio a la regularidad", es decir, aquel con mejor media en los tres lanzamientos.

32. Escribir un algoritmo que calcule la media aritmética de los elementos de una matriz de tamaño $m \times n$.

33. Escribir un algoritmo que permita multiplicar una matriz de reales de tamaño $m \times n$ y un vector de reales de tamaño n.

34. Escribir un algoritmo que permita multiplicar dos matrices de dimensiones compatibles, es decir, una matriz de de tamaño $m \times n$ con otra de tamaño $n \times p$.

35. Escribir un algoritmo que permita almacenar una matriz de tamaño $m \times n$ en un vector de p elementos.

36. En una matriz de reales de tamaño $m \times n$, se han almacenado las notas de las n asignaturas de m alumnos de primero. Es decir, en el elemento [i][j] se guarda la nota del alumno i en la asignatura j. Se pide:

    a) Un algoritmo que devuelva sobre un vector la nota media de cada alumno en el curso completo (definir el tipo de datos necesario para dicho parámetro de salida).

    b) Un algoritmo que devuelva sobre un vector la media de las notas de cada asignatura (definir el tipo de datos necesario para dicho parámetro de salida).

    c) Un algoritmo que devuelva los alumnos que han aprobado todas las asignaturas del curso completo (definir el tipo de datos necesario para los parámetros de salida).

37. Una matriz de tamaño $N \times N$ es la matriz identidad de tamaño N cuando todos sus elementos son 0 salvo los de la diagonal, que valen 1. Por ejemplo,

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Escribir una función que determine si una matriz dada de tamaño $N \times N$ es o no es la matriz identidad de tamaño N.

38. Escribe una función o procedimiento que devuelva cierto si, dada una matriz de enteros de `N` filas y `M` columnas, existe al menos una fila cuyos elementos sean todos iguales a uno dado, y falso en caso contrario.

39. Se quieren guardar los elementos no nulos de una matriz cuadrada de reales de tamaño $n \times n$ (definida como tipo `tMatriz`) en un vector de tamaño p (definido como tipo `tVector`). ¿Es correcto el siguiente algoritmo?¿Por qué? (Nota: hay p o menos elementos no nulos en la matriz)

```
void problematico(matriz A, int n, int p, vector v){
     int i, j, k;

     for(k=0; k<p;k++){
          for(i=0;i<n;i++){
               for(j=0;j<n;j++){
                    if (A[i][j]!=0.0)
                         v[k]=A[i][j];
               }
          }
     }
}
```

40. ¿Y este otro?¿Por qué?

```
void otroProblematico(matriz A, int n, int p, vector v){
     int i, j, k;

     k=1;
     for(i=0;i<n;i++){
          for(j=0;j<n;j++){
               if (A[i][j]!=0.0)
                    v[k]=A[i][j];
               k++;
          }
     }
}
```

41. Escribir un algoritmo que permita almacenar los elementos no nulos de una matriz de reales $n \times n$ en un vector, sabiendo que de cada elemento que se almacene, además del valor se quiere guardar el valor de sus índices.

42. Escribir un algoritmo que permita, dada una matriz de reales de tamaño $m \times n$, calcular la suma de su diagonal principal y de su diagonal secundaria. Ojo, que m y n pueden ser valores cualesquiera: iguales, distintos y puede ser que $m < n$ o que $m > n$.

    Por ejemplo, si la matriz fuera,

$$\begin{pmatrix} \mathbf{1,5} & 2,2 & 0,44 & 5,1 \\ 0,2 & \mathbf{3,3} & 7,5 & 8,25 \\ 0,33 & 4 & \mathbf{0,99} & 1,33 \\ 4,25 & 6,75 & 1,0 & \mathbf{2,1} \\ 0,56 & 7,25 & 1,1 & 33,5 \end{pmatrix}$$

    el resultado sería 7.89, suma de la diagonal principal, y 20.85, suma de la diagonal secundaria.

43. Se ha definido un tipo `tMatrizChar`, como un array de tipo base char con `N` filas y `M` columnas. En cada elemento se almacena una letra o blanco. Por ejemplo,

H	O	L	A									
A	D	I	O	S								
C	A	R	A	M	B	O	L	A				
E	N	T	R	E	N	A	M	I	E	N	T	O
Y	O											
S	O	B	R	A	D	A	M	E	N	T	E	

    Hay que escribir una función o procedimiento (justificar la elección) que procese la matriz y devuelva en un vector el número de elementos válidos en cada fila, es decir, cuántas letras hay antes de encontrar el blanco.

44. Escribir un algoritmo que dada una matriz $n \times n$, construya un vector v de n elementos, de forma que el valor de cada elemento v[i] se obtenga al sumar todos los elementos de la submatriz contenida entre las filas 0 e i y las columnas 0 e i. Por ejemplo, si

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} \\ a_{30} & a_{31} & a_{32} & a_{33} & a_{34} \\ a_{40} & a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

    entonces v[0]=a_{00}, en v[1] estaría la suma de los elementos

$$\begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix}$$

    y, en v[2],

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}$$

    y así, sucesivamente. El algoritmo sólo se considerará correcto si en el cálculo de cada elemento v[i] se saca provecho del resultado obtenido para el elemento anterior, v[i-1].

45. Suponiendo que se ha definido el tipo `tMatriz` como una matriz de tipo base real de tamaño $M \times N$:

a) Escribir una función o procedimiento (justificar la elección) que desplace los elementos de una fila genérica `f` una posición a la derecha, salvo el último elemento que se colocará como primero.

Por ejemplo,

Valor inicial matriz y `f` es 2:

$$\begin{pmatrix} 1{,}0 & 2{,}0 & 3{,}0 & 4{,}0 & 5{,}0 \\ 1{,}1 & 2{,}2 & 3{,}3 & 4{,}4 & 5{,}5 \\ 0{,}1 & 0{,}2 & 0{,}3 & 0{,}4 & 0{,}5 \\ 1{,}01 & 2{,}02 & 3{,}03 & 4{,}04 & 5{,}05 \end{pmatrix} \rightarrow$$

Valor final matriz:

$$\begin{pmatrix} 1{,}0 & 2{,}0 & 3{,}0 & 4{,}0 & 5{,}0 \\ 1{,}1 & 2{,}2 & 3{,}3 & 4{,}4 & 5{,}5 \\ \mathbf{0{,}5} & \mathbf{0{,}1} & \mathbf{0{,}2} & \mathbf{0{,}3} & \mathbf{0{,}4} \\ 1{,}01 & 2{,}02 & 3{,}03 & 4{,}04 & 5{,}05 \end{pmatrix}$$

b) Escribir una función o procedimiento (justificar la elección) que utilice la función o procedimiento del apartado anterior, para desplazar una posición a la derecha todas las columnas de una matriz.

Por ejemplo,

Valor inicial matriz:

$$\begin{pmatrix} 1{,}0 & 2{,}0 & 3{,}0 & 4{,}0 & 5{,}0 \\ 1{,}1 & 2{,}2 & 3{,}3 & 4{,}4 & 5{,}5 \\ 0{,}1 & 0{,}2 & 0{,}3 & 0{,}4 & 0{,}5 \\ 1{,}01 & 2{,}02 & 3{,}03 & 4{,}04 & 5{,}05 \end{pmatrix} \rightarrow$$

Valor final matriz:

$$\begin{pmatrix} 5{,}0 & 1{,}0 & 2{,}0 & 3{,}0 & 4{,}0 \\ 5{,}5 & 1{,}1 & 2{,}2 & 3{,}3 & 4{,}4 \\ 0{,}5 & 0{,}1 & 0{,}2 & 0{,}3 & 0{,}4 \\ 5{,}05 & 1{,}01 & 2{,}02 & 3{,}03 & 4{,}04 \end{pmatrix}$$

46. Se necesita escribir una pequeña aplicación para automatizar las reservas de un cine. Para ello, se ha comenzado representando las butacas como una tabla de 2 dimensiones (N filas y M columnas) en la que cada elemento puede valer:

    -1 : ese asiento no está disponible (no se puede vender la entrada porque falta una butaca, o lo que sea)

    0 : ese asiento ya está reservado (está vendida la entrada)

    1 : ese asiento está libre (entrada a la venta)

    Se pide:

    a) Escribir una función o procedimiento que cuente el número total de butacas libres.

    b) Escribir una función o procedimiento que, dado un número de asientos consecutivos que se desea reservar, `nAsientos`, y dada una posición de comienzo, `nFila` y `nColumna`, reserve las `nAsientos` butacas en la fila `nFila` desde la columna `nColumna` en adelante. Además, debe devolver un valor de tipo `boole` que indique si se ha podido realizar la reserva o no se ha podido realizar.

    c) Escribir una función o procedimiento que indique, por cada fila, cuántas butacas libres hay y cuántas butacas no disponibles hay.

47. Se quiere escribir un programa para realizar el proceso de las encuestas del profesorado. La nota media de un profesor para una asignatura se obtendrá procesando una matriz en la que cada una de las *f* filas representa la nota obtenida en cada pregunta de la encuesta y cada una de las *c* columnas la nota otorgada por un alumno.

    Para evitar respuestas subjetivas, la primera pregunta es lo que se llama un "medidor de objetividad". Por ejemplo, la pregunta "El profesor asiste a clase", debería tener una respuesta uniforme, puesto que el profesor acude a clase para todo el mundo o para nadie. Este medidor se utiliza para descartar respuestas subjetivas de la siguiente forma: Se calcula la media de esa pregunta y su desviación típica. Todas aquellas columnas en las que la primera pregunta no esté dentro del rango [media - desviación ... media + desviación], se descartan.

El algoritmo que permita calcular la media de un profesor, debe desechar las columnas no válidas. Se recuerda que la desviación típica de un vector se calcula como

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(a_i - \tilde{a})^2}{n}}$$

siendo $\tilde{a}$ el valor de la media.

48. A unas elecciones se presentan *NP* partidos que pretenden repartirse los *NE* escaños del Congreso de los Diputados.

    Escribir un programa que leerá un vector que contenga, por cada partido, el número de votos que ha obtenido y que imprima otro vector que contenga, por cada partido, el número de escaños que ha conseguido.

    El método a utilizar para asignar escaños a los partidos es el siguiente (ley de D'Hont simplificada):

    - Se construye una tabla de *NE* columnas y *NP* filas.
    - La primera columna coincide con los resultados en votos obtenidos por cada partido; las sucesivas, columna j con j=1...*NE-1*, se construyen dividiendo los valores de la primera columna entre (j+1) (división entera).
    - Los escaños se reparten atendiendo a las valores máximos de la tabla: el primer escaño al valor más grande, el segundo al siguiente más grande y así sucesivamente hasta repartir los *NE* escaños.