

## Prueba P1.1

Apellidos y nombre:.....

### Enunciado

Realizar una función que reciba como parámetros:

- 1) por referencia constante un vector de `string`, cada uno de los cuales consistirá únicamente en una serie de caracteres '1' y '0'.
- 2) un entero sin signo (pivote).

y devuelva un vector de enteros sin signo con aquellos que son **menores** que el pivote. Los enteros a considerar son los resultantes de convertir a entero la representación binaria que figura en cada uno de los `string`.

En el programa principal (`principal.cpp`) se leerán los `string` de un fichero cuyo nombre se pasará como primer parámetro al programa. El segundo parámetro será el pivote, si no está disponible valdrá 20.

El nombre del fichero deberá comenzar por "datos" y terminar por ".dat" si falta alguno se añadirá antes de acceder al fichero.

Tras invocar a la función, se mostrará, por salida estándar:

1. 'Pivote: ' seguido del valor considerado para el pivote.
2. 'Resultado: ' seguido de los elementos devueltos por la función junto con su representación en hexadecimal entre paréntesis precedida por '0x', en una sola línea separados por una coma y un espacio.

En caso de producirse algún error, el programa deberá devolver un código de error distinto de 0.

Ficheros de datos: `datosBinary1.dat` `datosBinarios2.dat` `datosBin3.dat`

### Guía de estilo

Al escribir el código fuente se debe de respetar las siguientes reglas:

- Indicar autor y fecha en la primera línea del fichero fuente.
- Usar dos espacios para cada nivel de sangrado, utilizando caracteres espacio.
- En cada línea no deberá haber más de una instrucción; aunque una instrucción puede ocupar varias líneas, en ese caso las líneas de continuación deberán tener dos niveles de sagrado más que la inicial (4 espacios más).
- No utilizar `using namespace std` sino especificar el espacio de nombre cuando sea necesario (`std::cout`, `std::endl`, `std::string`, `std::vector`, etc.)
- Dejar un espacio antes y después de cada operador, en especial de los de asignación (`=`, `+=`, `-=`, ...), comparación (`>`, `<`, `==`, ...) y los de envío y recepción de `stream` (`<<` y `>>`).
- Las líneas deberán tener, preferiblemente, menos de 72 caracteres y nunca superar los 80.
- Utilizar mensajes de depuración allí donde sea conveniente. Se utilizará la salida de error (`std::cerr`) para mostrar dichos mensajes.
- Declarar las variables justo antes de ser utilizadas y, si es necesario, hacer la declaración y la inicialización en la misma instrucción.

## Prueba P1.1

Apellidos y nombre:.....

### Enunciado

Realizar una función que reciba como parámetros:

- 3) por referencia constante un vector de `string`, cada uno de los cuales consistirá únicamente en una serie de caracteres '1' y '0'.
- 4) un entero sin signo (pivote).

y devuelva un vector de enteros sin signo con aquellos que son **mayores** que el pivote. Los enteros a considerar son los resultantes de convertir a entero la representación binaria que figura en cada uno de los `string`.

En el programa principal (`principal.cpp`) se leerán los `string` de un fichero cuyo nombre se pasará como primer parámetro al programa. El segundo parámetro será el pivote, si no está disponible valdrá 16.

El nombre del fichero deberá comenzar por "datos" y terminar por ".dat" si falta alguno se añadirá antes de acceder al fichero.

Tras invocar a la función, se mostrará, por salida estándar:

1. 'Pivote: ' seguido del valor considerado para el pivote.
2. 'Resultado: ' seguido de los elementos devueltos por la función junto con su representación en hexadecimal entre paréntesis precedida por '0x', en una sola línea separados por una coma y un espacio.

En caso de producirse algún error, el programa deberá devolver un código de error distinto de 0.

Ficheros de datos: `datosBinary1.dat` `datosBinarios2.dat` `datosBin3.dat`

### Guía de estilo

Al escribir el código fuente se debe de respetar las siguientes reglas:

- Indicar autor y fecha en la primera línea del fichero fuente.
- Usar dos espacios para cada nivel de sangrado, utilizando caracteres espacio.
- En cada línea no deberá haber más de una instrucción; aunque una instrucción puede ocupar varias líneas, en ese caso las líneas de continuación deberán tener dos niveles de sagrado más que la inicial (4 espacios más).
- No utilizar `using namespace std` sino especificar el espacio de nombre cuando sea necesario (`std::cout`, `std::endl`, `std::string`, `std::vector`, etc.)
- Dejar un espacio antes y después de cada operador, en especial de los de asignación (`=`, `+=`, `-`, `=`, ...), comparación (`>`, `<`, `==`, ...) y los de envío y recepción de `stream` (`<<` y `>>`).
- Las líneas deberán tener, preferiblemente, menos de 72 caracteres y nunca superar los 80.
- Utilizar mensajes de depuración allí donde sea conveniente. Se utilizará la salida de error (`std::cerr`) para mostrar dichos mensajes.
- Declarar las variables justo antes de ser utilizadas y, si es necesario, hacer la declaración y la inicialización en la misma instrucción.

**Prueba P1.1**

Apellidos y nombre:.....

**Enunciado**

Realizar una función que reciba como parámetros:

- 1) por referencia constante un vector de enteros.
- 2) un entero sin signo ( $n$ ).
- 3) un entero sin signo ( $m$ ).

y devuelva un vector de `bool` que corresponderá al valor del bit  $m$ -simo (en la posición  $m$  de la representación binaria del entero) de los elementos en posición múltiplo de  $n$  del vector de enteros.

En el programa principal (`principal.cpp`) se leerán los datos de un fichero cuyo nombre se pasará como primer parámetro al programa. El segundo parámetro será el valor del bit a considerar ( $m$ ). El tercer parámetro será el parámetro  $n$  de la función, si no está disponible valdrá 2.

El nombre del fichero deberá comenzar por “datos” y terminar por “.dat” si falta alguno se añadirá antes de acceder al fichero.

El vector devuelto por la función se escribirá en fichero que tenga el mismo nombre que el entrada (sin el .dat) pero terminado en '`_m_n.dat`', donde  $m$  y  $n$  serán la representación decimal las variables del mismo nombre. Se escribirá en cada línea del fichero 'Verdad', 'Falso' según corresponda.

En caso de producirse algún error, el programa deberá devolver un código de error distinto de 0.

Ficheros de datos: `datosEnteros1.dat` `datosInt2.dat` `datosInteg3.dat`

**Guía de estilo**

Al escribir el código fuente se debe de respetar las siguientes reglas:

- Indicar autor y fecha en la primera línea del fichero fuente.
- Usar dos espacios para cada nivel de sangrado, utilizando caracteres espacio.
- En cada línea no deberá haber más de una instrucción; aunque una instrucción puede ocupar varias líneas, en ese caso las líneas de continuación deberán tener dos niveles de sangrado más que la inicial (4 espacios más).
- No utilizar `using namespace std` sino especificar el espacio de nombre cuando sea necesario (`std::cout`, `std::endl`, `std::string`, `std::vector`, etc.)
- Dejar un espacio antes y después de cada operador, en especial de los de asignación (`=`, `+=`, `-=`, ...), comparación (`>`, `<`, `==`, ...) y los de envío y recepción de stream (`<<` y `>>`).
- Las líneas deberán tener, preferiblemente, menos de 72 caracteres y nunca superar los 80.
- Utilizar mensajes de depuración allí donde sea conveniente. Se utilizará la salida de error (`std::cerr`) para mostrar dichos mensajes.
- Declarar las variables justo antes de ser utilizadas y, si es necesario, hacer la declaración y la inicialización en la misma instrucción.

## Prueba P1.1

Apellidos y nombre:.....

### Enunciado

Realizar una función que reciba como parámetros:

- 1) por referencia constante un vector de enteros.
- 2) un entero sin signo ( $m$ ).
- 3) un entero (*pivote*).

y devuelva un vector de enteros con los elementos que, teniendo su  $m$ -simo bit a 1, sean mayores que el *pivote*.

En el programa principal (`principal.cpp`) se leerán los datos de un fichero cuyo nombre se pasará como primer parámetro al programa. El segundo parámetro será el número del bit a considerar ( $m$ ). El tercer parámetro será el *pivote*, si no está disponible valdrá 0.

El nombre del fichero deberá comenzar por “datos” y terminar por “.dat” si falta alguno se añadirá antes de acceder al fichero.

El vector de elementos mayores se escribirá en fichero que tenga el mismo nombre que el entrada (sin el .dat) pero terminado en '`_m_mayores_pivote.dat`', donde  $m$  y *pivote* será la representación decimal de las variables correspondientes. Se escribirá en cada línea del fichero un entero y, separado por un espacio y entre paréntesis, su representación hexadecimal precedida por '`0x`'.

En caso de producirse algún error, el programa deberá devolver un código de error distinto de 0.

Ficheros de datos: `datosEnteros1.dat` `datosInt2.dat` `datosInteg3.dat`

### Guía de estilo

Al escribir el código fuente se debe de respetar las siguientes reglas:

- Indicar autor y fecha en la primera línea del fichero fuente.
- Usar dos espacios para cada nivel de sangrado, utilizando caracteres espacio.
- En cada línea no deberá haber más de una instrucción; aunque una instrucción puede ocupar varias líneas, en ese caso las líneas de continuación deberán tener dos niveles de sangrado más que la inicial (4 espacios más).
- No utilizar `using namespace std` sino especificar el espacio de nombre cuando sea necesario (`std::cout`, `std::endl`, `std::string`, `std::vector`, etc.)
- Dejar un espacio antes y después de cada operador, en especial de los de asignación (`=`, `+=`, `-=`, ...), comparación (`>`, `<`, `==`, ...) y los de envío y recepción de `stream` (`<<` y `>>`).
- Las líneas deberán tener, preferiblemente, menos de 72 caracteres y nunca superar los 80.
- Utilizar mensajes de depuración allí donde sea conveniente. Se utilizará la salida de error (`std::cerr`) para mostrar dichos mensajes.
- Declarar las variables justo antes de ser utilizadas y, si es necesario, hacer la declaración y la inicialización en la misma instrucción.