Observaciones

- Se disponen de 3 horas para realizar los ejercicios
- Los resultados se harán públicos en la página web de la Escuela Politécnica Superior de la Universidad de Alicante (https://eps.ua.es/) antes del día 12 de abril

Problema 1: Calculador de números primos (3 puntos)

Un número primo es un número entero que sólo es divisible por sí mismo y por `1`. Es decir, si no hay ningún número entre `1` (sin incluir a éste) y `n` que pueda dividir a `n` sin resto, entonces `n` es primo.

Se pide realizar un programa que calcule los números primos que se encuentran en el rango desde `2` hasta `m`. El programa debe pedir un número máximo `m` e imprimir todos los primos en el rango indicado.

Ejemplo de funcionamiento

```
Introduzca el valor m: 21
Números primos: 2 3 5 7 11 13 17 19
```

Problema 2: Comprobador de ISBN (3 puntos)

Un ISBN es un código de `10` dígitos que identifica de forma única a un libro. Los primeros `9` dígitos representan el libro y el último dígito se utiliza para comprobar que el ISBN es correcto.

Para comprobar que un ISBN es correcto se debe multiplicar el primer dígito por `10`, el segundo por `9`, el tercero por `8` y así sucesivamente hasta que el último dígito se multiplica por `1`. Si el número resultante es un múltiplo de `11`, el código ISBN es válido. Es posible que al número resultante haya que sumarle `10` para que el ISBN sea válido. En ese caso se usa la letra `X`.

Por ejemplo, `0201103311` es un ISBN válido porque:

```
10*0 + 9*2 + 8*0 + 7*1 + 6*1 + 5*0 + 4*3 + 3*3 + 2*1 + 1*1 = 55
```

y `55` es múltiplo de `11` (`55 = 11*5`).

Otro ejemplo de número ISBN válido es `156881111X`:

```
10*1 + 9*5 + 8*6 + 7*8 + 6*8 + 5*1 + 4*1 + 3*1 + 2*1 + 1*10 = 231
```

y `231` es múltiplo de `11` (`231 = 11*21`).

Escribe un programa que compruebe si un ISBN es válido.

Ejemplo de funcionamiento

```
Introduzca ISBN: 0201103311
El ISBN es válido

Introduzca ISBN: 0201103411
El ISBN no es válido
```

Añade una opción al programa anterior que permita reparar números ISBN cuando detecte que le falta un dígito (marcado con el símbolo `?`). En ese caso, el programa debe imprimir el valor correcto del dígito que falta.

Ejemplo de funcionamiento

```
Introduzca ISBN: 15688?111X
El dígito que falta es 1
```

Problema 3: Sopa de letras (4 puntos)

Dada una matriz de `m * n` letras y una lista de palabras, realizar un programa que encuentre la ubicación en la matriz en la que se puede encontrar cada una de las palabras de la lista.

Una palabra debe coincidir con una línea recta e ininterrumpida de letras en la matriz. Las letras mayúsculas y minúsculas se consideran equivalentes. Es decir, si buscamos `sopa`, podemos considerar `Sopa` o `sOPa` como coincidencias válidas. La coincidencia se puede dar en cualquiera de las cuatro direcciones horizontales y verticales (las coincidencias diagonales no se deben tener en cuenta).

El programa recibirá una entrada con el siguiente formato:

- La primera línea incluirá un par de números enteros `m` y `n`, donde `m>=1` y `n<=50`
- Las siguientes `m` líneas contendrán `n` letras cada una, representando la matriz de letras donde deben buscarse las palabras. Las letras en la matriz pueden estar en mayúsculas o minúsculas
- A continuación de la matriz de letras, aparecerá un número entero `k`, donde `1<=k<=20`
- Las siguientes `k` líneas de entrada contienen la lista de palabras a buscar, una por cada línea. Estas palabras sólo pueden contener letras mayúsculas y minúsculas (sin espacios, guiones u otros caracteres no alfabéticos)

Para cada palabra, el programa generará un par de números enteros que representen su ubicación en la matriz. Los números enteros estarán separados por un solo espacio:

- El primer entero es la línea de la matriz donde se puede encontrar la primera letra de la palabra dada (`1` representa la fila superior de la matriz y `m` representa la fila inferior)
- El segundo entero es la columna de la matriz donde se puede encontrar la primera letra de la palabra dada (`1` representa la columna más a la izquierda de la matriz y `n` representa la columna más a la derecha)

Si una palabra se puede encontrar más de una vez en la matriz, se debe devolver la ubicación de la ocurrencia más alta de la palabra, es decir, aquella que sitúa la primera letra de la palabra más cerca de la parte superior izquierda de la matriz. Se asumirá que todas las palabras de la lista de entrada se encuentran al menos una vez en la matriz.

Ejemplo de funcionamiento

Entrada al programa	Salida generada por el programa

`8 11`
`abcDEFGhigg`
`hEbkWalDorf`
`FtaAwaldORm`
`FtmimrLqsrc`
`byBArBeTTYv`
`KlIbqwikomk`
`strEBGadhrb`
`yUiqlxcnBjk`
`4`
`Waldorf`
`Bambi`
`Betty`
`Dagbert`
	`2 5`
	`2 3`
	`1 2`
	`7 8`