UNIVERSIDAD DE MÁLAGA
Dpto. Lenguajes y Ciencias de la Computación
E.T.S.I. Informática

Fundamentos de la Programación
Examen 1ª Convocatoria Ordinaria **01/02/19**

Apellidos, Nombre:

Titulación:

Grupo:

Código PC usado:

NOTAS PARA LA REALIZACIÓN DEL EXAMEN:

- La solución se almacenará en la carpeta **EXAMENFEBFP**, dentro de **Documentos**. Si la carpeta ya existe, debe borrarse todo su contenido. En otro caso, debe crearse.
- Los nombres de los ficheros con la solución para los ejercicios 1, 2, 3 y 4 serán **ejercicio1.cpp**, **ejercicio2.cpp, ejercicio3.cpp** y **ejercicio4.cpp,** respectivamente.
- Al inicio del contenido de cada fichero deberá aparecer un comentario con **el nombre del alumno, titulación, grupo y código del equipo** que se está utilizando (cada dato en una línea diferente)**.**
- **Debe consultarse** el documento "**Obligaciones y Recomendaciones Estilo de Programación**", disponible en el Campus Virtual de la asignatura, con objeto de tener en cuenta los puntos allí señalados en las soluciones a los ejercicios.
- Una vez terminado el examen, se subirán los ficheros ***.cpp** a la tarea creada en el **campus virtual** para ello.
- **No está permitido**:
  - Utilizar documentación electrónica o impresa.
  - Intercambiar documentación con otros compañeros.

(**1 pto) 1.- Diseña** un **procedimiento** *menorEstricto* con tres parámetros. El primero es de entrada para recibir un array completo de tamaño TAM (una constante establecida) de números enteros. El segundo parámetro es de salida e indicará si se ha encontrado el valor menor estricto (menor único) en el array o no. El tercer parámetro es de salida y será ese valor menor estricto en caso de que exista.

Importante: sólo se completará el código del procedimiento *menorEstricto* en el fichero *ejercicio1.cpp* proporcionado en el campus virtual. Puedes añadir más procedimientos o funciones si lo estimas necesario. No se debe modificar el resto del código proporcionado. La puntuación de este problema será de 1 punto sólo en el caso de que el procedimiento funcione correctamente. En otro caso la puntuación será de 0 puntos.

La ejecución del código suministrado (tras diseñar el procedimiento solicitado) será:

```
El menor estricto del primer array es: 3
El segundo array no tiene menor estricto
El menor estricto del tercer array es: 2
```

(**1.5 ptos) 2.-** Un antiguo método de cifrado se basa en el uso de una clave alfanumérica, compuesta por **todas** las letras minúsculas (sin incluir la letra eñe) y **todos** los dígitos, dispuesta en una tabla bidimensional de 6×6 elementos como en el siguiente ejemplo:

	0	1	2	3	4	5
0	p	k	a	f	5	v
1	e	o	9	t	y	0
2	s	3	z	7	d	j
3	r	b	n	u	m	1
4	2	w	4	h	8	g
5	c	x	6	q	i	l

Diseña un **procedimiento** *cifrar* con tres parámetros. El primero es de entrada para recibir una clave almacenada en una tabla de cifrado como la especificada anteriormente. El segundo parámetro también es de entrada para recibir una cadena de caracteres conteniendo el texto a cifrar. El tercer parámetro es de salida y será una cadena de caracteres conteniendo el texto cifrado. El procedimiento se realiza de la siguiente forma:

- Para cada **par** de caracteres del texto de entrada (texto a cifrar), se producen también **dos** caracteres en el texto cifrado, según el siguiente proceso:
  - Se buscan ambos caracteres en la tabla de la clave (*se puede suponer que los dos caracteres estarán en la tabla*), devolviendo los índices de la fila y columna donde está almacenado cada carácter (*f1*, *c1*) y (*f2*, *c2*). Por ejemplo, usando la clave mostrada anteriormente, para el par de caracteres `ab`, se obtienen las coordenadas (*0*, *2*) y (*3*, *1*) de la tabla.
  - Se obtienen los caracteres que se encuentran en la tabla en las posiciones (*f1*, *f2*) y (*c1*, *c2*). Siguiendo con el ejemplo de clave anterior, en estas coordenadas se encuentran los caracteres `f3` según la tabla.
  - Estos nuevos caracteres se añaden al final del texto cifrado (tercer parámetro).
- Se repite el proceso anterior por cada par de caracteres del texto de entrada. En caso de que el texto de entrada tenga un número de caracteres impar, se desecha el último carácter del mismo, y no será tenido en cuenta.
- Se puede suponer que en el texto de entrada sólo aparecerán letras minúsculas y dígitos. También se puede suponer que la tabla clave también es correcta (aparecen todas las letras minúsculas y todos los dígitos).

Importante: sólo se completará el código del procedimiento *cifrar* en el fichero *ejercicio2.cpp* proporcionado en el campus virtual. Puedes añadir más procedimientos o funciones si lo estimas necesario. No se debe modificar el resto del código proporcionado.

La ejecución del código suministrado (tras diseñar el procedimiento) será:

```
El texto cifrado es: wbc6e4j89e
```

(**3.5 ptos) 3.- Diseña** un **algoritmo** que lea de teclado un texto y muestre por pantalla un listado de todas las palabras del texto indicando para cada una su primera y última posición en el texto. **En la salida no habrá palabras repetidas**.

Ejemplo:

Entrada:

```
Introduzca un texto (FIN para terminar):
CREO QUE IREMOS A MI CASA PRIMERO Y QUE DESPUES IREMOS A LA CASA QUE
QUIERAS FIN
```

Salida:

```
Palabras y posiciones primera y última:
CREO 1 1
QUE 2 15
IREMOS 3 11
A 4 12
MI 5 5
CASA 6 14
PRIMERO 7 7
Y 8 8
DESPUES 10 10
LA 13 13
QUIERAS 16 16
```

NOTAS:

- El texto contiene un número indefinido de palabras.
- El texto termina con la palabra FIN.
- Cada palabra tiene un número indefinido pero limitado de caracteres (todos alfabéticos mayúsculas).
- El carácter separador de palabras es el espacio en blanco.
- En el texto habrá un número máximo MAX_PAL_DIST (una constante) de palabras distintas.

(**4 ptos) 4.- Diseña** un **algoritmo** que lea de teclado

- un número natural k
- y una colección de valores enteros para completar una matriz de tamaño $N \times M$ (siendo N y M dos constantes definidas en el programa).

A partir de los datos leídos, el algoritmo calculará y mostrará por pantalla los k valores de la matriz que más veces se repiten.

Notas:

- Si k es mayor que el número de valores distintos de la matriz, entonces se mostrarán todos esos valores distintos (ver ejemplo 2).
- Si hay valores de la matriz que se repiten igual número de veces y alguno de ellos no se puede mostrar porque de hacerlo se excedería el valor de k, da igual los valores que queden sin mostrarse (ver ejemplo 3).

Ejemplos:

Ejemplo 1)

Entrada:

```
Introduzca k: 4
Introduzca la matriz 3x4 (por filas):
45 -17 867 45
2 867 -17 3
1 -2 45 3
```

Salida:

```
Los valores que mas se repiten son: 45 -17 867 3
```

Explicación:

Ya que para k = 4 y la matriz (N = 3, M=4):

45	-17	867	45
2	867	-17	3
1	-2	45	3

Se mostrarán los números (da igual el orden): 45 -17 867 3

Ejemplo 2)

Entrada:

```
Introduzca k: 8
Introduzca la matriz 3x4 (por filas):
45 -17 867 45
2 867 -17 3
1 -2 45 3
```

Salida:

```
Los valores que mas se repiten son: 45 -17 867 3 2 1 -2
```

Explicación:

Ya que para la misma matriz y k = 8, se mostrarían los números (da igual el orden): 45 -17 867 3 2 1 -2

(Aunque se pide mostrar 8 números, sólo se muestran 7, los que hay)

Ejemplo 3)

Entrada:

```
Introduzca k: 3
Introduzca la matriz 3x4 (por filas):
45 -17 867 45
2 867 -17 3
1 -2 45 3
```

Salida:

```
Los valores que mas se repiten son: 45 -17 867
```

Explicación:

Ya que para la misma matriz y k = 3, se mostrarían los números (da igual el orden): 45 -17 867

(También podrían mostrarse: 45 -17 3) (También podrían mostrarse: 45 867 3) …