Nombre y Apellido: ..... N° Legajo: .....

Recuperatorio Primer Parcial de Programación Imperativa

24/06/2022

- **<u>Condición mínima de aprobación: Sumar 5 (cinco) puntos.</u>**
- **Los ejercicios que no se ajusten estrictamente al enunciado, <u>no serán aceptados</u>.**
- **No usar variables globales ni static.**
- **No es necesario escribir los #include**

Ejercicio 1 (3,5 puntos)

Los teléfonos (incluyendo los celulares) poseen grabado en sus teclas no sólo el número sino un conjunto de letras, esto quiere decir que **a cada dígito (excepto el 0 y el 1) le corresponden 3 posibles letras**. A continuación, se colocan las equivalencias de un teléfono tradicional entre el valor numerico y las letras.

- Tecla 0: no posee valores alfabéticos
- Tecla 1: no posee valores alfabéticos
- Tecla 2: ABC
- Tecla 3: DEF
- Tecla 4: GHI
- Tecla 5: JKL
- Tecla 6: MNO
- Tecla 7: PQRS
- Tecla 8: TUV
- Tecla 9: WXYZ

Hacer una función void **`convertirATexto`** que reciba como parámetros:

- Un **unsigned long** representando un número teléfonico
- Un **vector de chars** con espacio suficiente para dejar una posible representación alfanumérica del número de teléfono

En caso que la tecla no tenga equivalencia, debe colocar el dígito original.

<u>Ejemplos:</u> **(los resultados son dependientes de la generación de números aleatorios)**

```
int main(void) {
    char s[10];
    srand(time(NULL));
    convertirATexto(0, s);
    assert(strcmp(s,"0") == 0);  // Única posible solución
    convertirATexto(1, s);
    assert(strcmp(s,"1") == 0);
    convertirATexto(101, s);
    assert(strcmp(s,"101") == 0);
    convertirATexto(10000, s);
    assert(strcmp(s,"10000") == 0);
    convertirATexto(23721, s);
    assert(strlen(s) == 5);
    assert(s[0]=='A' || s[0] == 'B' || s[0]=='C');
    assert(s[1]=='D' || s[1] == 'E' || s[1]=='F');
    assert(s[2]=='P' || s[2] == 'Q' || s[2]=='R' || s[2]=='S');
```

```
    assert(s[3]=='A' || s[3] == 'B' || s[3]=='C');
    assert(s[4]=='1');
    return 0;
}
```

Ejercicio 2 (3 puntos)

Se desea eliminar de un texto todas las vocales, siempre y cuando **no sean vocales "sueltas"**, esto es, para eliminarlas deben tener al menos **una letra antes o una letra después (y que no sea vocal)**.

Escribir una función **`eliminaVoc`** que reciba únicamente un string y elimine todas las vocales siguiendo las reglas mencionadas anteriormente.

Ejemplos:

- Si recibe `"hola mundo"` lo deja como `"hl mnd"`
- Si recibe `"hola a todo el mundo"` lo deja como `"hl a td l mnd"`
- Si recibe `"xyz"` lo deja sin cambios
- Si recibe el string `"aholoaaa"` lo deja como `"hlaaa"`
- Si recibe el string vacío `""` lo deja sin cambios
- Si recibe `"aa eo iu oa uo"` lo deja sin cambios
- Si recibe `"pa po pi po pu"` lo deja como `"p p p p p"`
- Si recibe `"aeiou"` lo deja igual
- Si recibe `"a,e,i,o,u"` lo deja igual

Ejercicio 3 (3,5 puntos)

Escribir una función **`esMagica`** que reciba como único parámetro una matriz de enteros de N filas y N columnas y **determine si es un "cuadrado mágico"**.

Una matriz es cuadrado mágico cuando:

- es cuadrada (NxN) (donde N es una constante simbólica)
- tiene **todos** los números del 1 al N^2
- la suma de los elementos de cualquier fila es igual a la suma de los elementos de cualquier columna

Ejemplos:

Para la siguiente matriz de dimensión 4x4 debe retornar 1, pues **es mágica** (están todos los números del 1 al 16 y los elementos de cada fila y cada columna suman 34):

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

La siguiente matriz de dimensión 4x4 **no es mágica** pues si bien están todos los números del 1 al 16, la primera fila suma 35 y la segunda 22:

16	4	5	10
11	2	3	6
12	9	13	15
7	8	1	14

La siguiente matriz de dimensión 3x3 **es mágica** pues están todos los números del 1 al 9 y los elementos de cada fila y cada columna suman 15:

4	9	2

3	5	7
8	1	6

La siguiente matriz de dimensión 3x3 **no es mágica** pues no están todos los números del 1 al 9 (faltan el 3 y el 8) a pesar de que los elementos de cada fila y cada columna suman 15:

4	9	2
4	5	6
7	1	7