Ejercicios de Introducción a la Programación en C

Nota importante: Todos los programas deben trabajar con argumentos que se pasan desde la línea de comandos al ejecutar el programa. Es decir, no se debe pedir entrada al usuario durante la ejecución, sino que los datos deben ser recibidos como argumentos (argv).

Hola Mundo

- Escribe un programa en C que imprima en pantalla el mensaje "Hola mundo".
- Este ejercicio te ayudará a familiarizarte con el proceso de compilación y ejecución.
- Compílalo usando gcc (o el compilador que uses) y ejecútalo desde un terminal. Por ejemplo:

  ```
  gcc -o hola -Wall -Wshadow -Wvla hola.c
  ```

- Ejecuta el programa en un terminal:

  ```
  ./hola
  ```

Imprimir argumentos

- Crea un programa que imprima todos los argumentos que recibe:
  a) Cada argumento en una línea diferente.
  b) Todos los argumentos en una sola línea, separados por espacios.
- Este ejercicio te enseña a recorrer el array argv.

Par o impar

- Crea un programa que reciba un número como argumento y determine si es par o impar.
- El programa debe imprimir un mensaje indicando el resultado.

Estadísticas básicas

- Escribe un programa que reciba varios números como argumentos.
- El programa debe imprimir:
  - El número más grande.
  - El número más pequeño.
  - La mediana (el valor central cuando están ordenados, si el número de elementos es par, debe ser la media de los dos elementos centrales).

Factorial

- Crea un programa que reciba un número como argumento y calcule su factorial.
- Ejemplo:

  ```
  ./programa 5
  ```

- La salida sería:

  ```
  120
  ```

Binario a decimal

- Escribe un programa que reciba un número en binario (como cadena) y lo convierta a decimal.
- Ejemplo:

  ```
  ./programa 1110
  ```
- La salida sería:

  ```
  14
  ```

Decimal a hexadecimal

- Crea un programa que reciba un número decimal y lo convierta a hexadecimal.
- Ejemplo:

  ```
  ./programa 255
  ```
- La salida sería:

  ```
  FF
  ```

Hexadecimal a decimal

- Escribe un programa que reciba un número en hexadecimal y lo convierta a decimal.
- Ejemplo:

  ```
  ./programa FF
  ```
- La salida sería:

  ```
  255
  ```

Array invertido (Números)

- Crea un programa que recibe varios números como argumentos.
- Guarda los números en un array y luego los imprime en orden inverso.

Array invertido (Palabras)

- Similar al anterior, pero con palabras en lugar de números.
- Guarda las palabras en un array y las imprime en orden inverso.

Ordenar palabras (Insertion Sort)

- Escribe un programa que recibe varias palabras.
- Después, guarda las palabras en un array y las ordena alfabéticamente usando el algoritmo insertion sort (ver la wikipedia).

Pila con array

- Escribe un programa que implemente una pila utilizando un array.
- El programar debe guardar las palabras recibidas en la pila e imprimirlas en orden inverso (como se sacan de la pila).

Pila con lista enlazada

- Similar al anterior, pero el programa implementa la pila usando una lista enlazada en lugar de un array.

Eliminar letra

- Escribe un programa que reciba varias palabras y una letra como argumentos.
- El programa debe imprimir las palabras con esa letra eliminada.

Invertir palabras

- Crea un programa que reciba varias palabras.
- El programa debe imprimir cada palabra invertida (por ejemplo, "hola" → "aloh").

Palíndromos

- Crea un programa que reciba varias palabras.
- El programa debe imprimir solo aquellas que sean palíndromos (palabras que se leen igual al derecho y al revés, como "oso" o "reconocer").

Ordenar estudiantes por nota

- Crea un programa que reciba información de estudiantes.
- Cada estudiante debe tener: nombre, DNI y nota.
- Define un tipo de datos para los estudiantes, con un `struct` y un `typedef`
- El programa debe ordenar a los estudiantes por nota (de mayor a menor) y mostrarlos ordenados.

Calculadora simple

- Escribe un programa que reciba tres argumentos: dos números y un operador (+, -, *, /).
- El programa debe realizar la operación indicada y mostrar el resultado.
- Ejemplo:

  ```
  ./programa 5 3 +
  ```

- Su salida debería ser:

  ```
  8
  ```

ASCII

- Escribe un programa que muestre por pantalla la lista de los caracteres ASCII del 32 al 126 (número + carácter), a 8 caracteres por línea, con el siguiente formato:

```
32:      33: !   34: "   35: #   36: $   37: %   38: &   39: '
40: (   41: )   42: *   43: +   44: ,   45: -   46: .   47: /
48: 0   49: 1   50: 2   51: 3   52: 4   53: 5   54: 6   55: 7
56: 8   57: 9   58: :   59: ;   60: <   61: =   62: >   63: ?
64: @   65: A   66: B   67: C   68: D   69: E   70: F   71: G
72: H   73: I   74: J   75: K   76: L   77: M   78: N   79: O
80: P   81: Q   82: R   83: S   84: T   85: U   86: V   87: W
88: X   89: Y   90: Z   91: [   92: \   93: ]   94: ^   95: _
```

```
 96: `   97: a   98: b   99: c  100: d  101: e  102: f  103: g
104: h  105: i  106: j  107: k  108: l  109: m  110: n  111: o
112: p  113: q  114: r  115: s  116: t  117: u  118: v  119: w
120: x  121: y  122: z  123: {  124: |  125: }  126: ~
```

- Es importante que no te escribas en tu programa explícitamente la lista de caracteres, lo deberías hacer con bucles.

Hexadecimal

- Escribe un programa que muestre por pantalla la representación hexadecimal de los bytes que componen la siguiente estructura:
  - el entero 27
  - el carácter 'z'
  - un array de 3 enteros: el 1000, el 10000, y el 100000
- La salida debe tener un formato similar a este:

```
0F FE 00 07 ...
```

- Nota: Prueba %x, %X, %02x, %02X, y mira la página de manual de así:

```
man 3 printf
```

Frecuencia

- Escribe un programa que haga un análisis de frecuencias de cada carácter de todos los argumentos, guarde los resultados en una estructura de datos, y muestre luego el resultado.
- Ejemplo: dada la entrada:

```
./freq hola que tal estamos
```

- Para ese ejemplo, la salida debería ser similar a:

```
a 3
e 2
h 1
l 2
m 1
o 2
q 1
s 2
t 2
u 1
```

Dígitos

- Escribe un programa que muestre todos los argumentos que contengan algún dígito.
- Nota: puedes usar la función de la biblioteca estándar de C llamada isdigit:

```
man 3 isdigit
```

Echo

- Escribe un programa que se comporte como el comando `echo` de Linux, pero interpretando los caracteres de escape

y tal y como actua echo con la opcion -e en el sistema operativo Linux).

```
man 1 echo
```

Cifrado de César

- Escribe un programa que cifre y descifre todos los argumentos menos el primero, utilizando como clave de cifrado y descifrado el valor numérico del primero, al que llamaremos N.
- El cifrado será de desplazamiento circular: de cada letra de la 'a' a la 'z' (minúsculas) que vayamos a cifrar mostraremos otra letra que esté desplazada N unidades.
- Por ejemplo, si N=3, "zapato" pasaría a ser "cdsdwr". Con N=-3, "cdsdwr" pasaría a ser "zapato". El programa debe admitir como clave un número negativo.
- Ejemplo:

```
./cesar 3 Es cara la cacatua.
```

- Su salida sería:

```
Ev fdud od fdfdwxd.
```

- Ejemplo:

```
./cesar 13 Efgb :-) rf MUY qviregvqb
```

- Su salida sería:

```
Esto :-) es MUY divertido
```

Binario

- Escribe un programa en C llamado "char2bin" que muestre la representación en binario de los argumentos del programa
- Ejemplo:

```
./char2bin hola amigos
```

- Su salida sería:

```
01100110011011110110111100100000011000100110000101110010
```

Binario a letras

- Escribe un programa que muestre los caracteres que se correspondan con los números en binario que se pasen como argumentos
- Ejemplo:

```
./bin2char 01100110011011110110111100100000011000100110000101110010
```

- Su salida sería:

```
hola amigos
```

Extensión

- Escribe un programa que muestre la extensión de los argumentos que se le pasen (esto es, los caracteres que aparezcan después del último punto, si lo hay)
- Ejemplo:

```
./extension uno.dos.tres caracola feo.holacomotamos
```

- La salida sería:

```
tres
holacomotamos
```

Imprimir los más largos

- Escribe un programa que muestre los 5 argumentos más largos.
- Si se le pasan menos de 5 argumentos, el programa debe dar un error.
- Si hay empate, da igual qué argumentos se escojan.
- Ejemplo:

```
./longargs x uno dos tres cuatro pepe pepepep popopo papapapa xx
```

- Su salida sería:

```
tres
cuatro
pepepep
popopo
papapapa
```