

Programación de software de sistemas

Ayudantía 1

Profesor: Rodrigo Verschae

Ayudante: Nicolás Araya

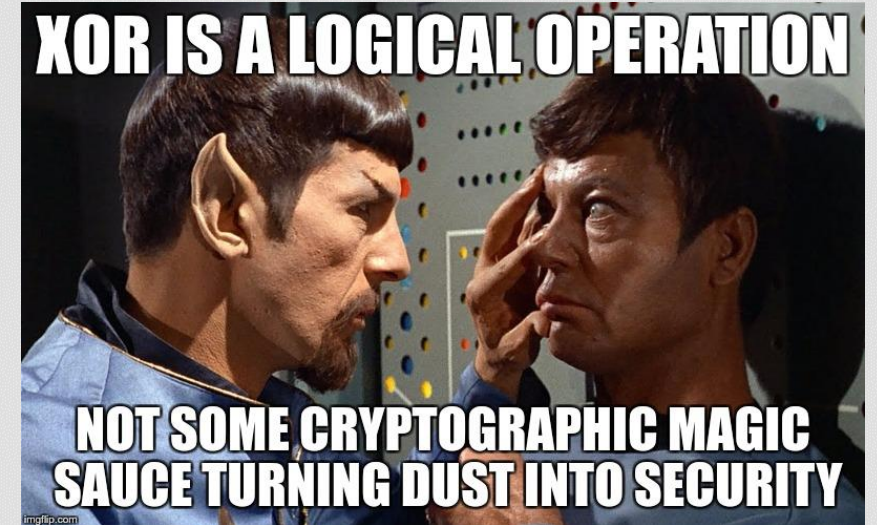


Calentamiento con XOR ^

`int` 222 => 1 1 0 1 1 1 1 0

`int` 111 => 0 1 1 0 1 1 1 1

`int` 177 => 1 0 1 1 0 0 0 1



Pregunta 1

Programe la función “`int XOR(int n, int m)`” que retorna “`n XOR m`”.

Hint: Puede descubrir la formula utilizando una tabla de verdad.



Solución

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10 int XOR(int n, int m)  
11 {  
12  
13  
14  
15     return (~n & m) | (n & ~m);  
16  
17  
18 }  
19  
20
```

```
21  
22  
23 int XOR(int n, int m)  
24 {  
25  
26  
27  
28     return ~(n & m) & (n | m);  
29  
30 }  
31  
32  
33
```



Bit más significativo.

¿Cuál sería el bit más significativo del entero 14?

`int` 14 => **1** 1 1 0
 $2^3 \ 2^2 \ 2^1 \ 2^0 = 8 + 4 + 2 = 14$

¿Cómo contamos los bits en C? ¿`int` `int_size(int n)`?



Pregunta 2.1

Cree la función “`unsigned int bit_mas_significativo(unsigned int n)`” que retorna una máscara de bits con ceros, salvo en la posición donde está el bit más significativo de `n`.

Si `n == 0110101` entonces `mask = 0100000`



Solución

```
1
2
3
4
5
6
7
8
9 int int_size(int n)
10 {
11     int c = 0;
12
13     while(n)
14     {
15         n = n >> 1;
16         c++;
17     }
18
19     return c;
20
21
22
23
24
25
26
27
28
29
30 }
31
32
33
```

```
unsigned int bit_mas_significativo(unsigned int n)
{
    unsigned int mask = n >> int_size(n)-1;
    return mask << int_size(n)-1;
}
```



Pregunta 2.2

Usando la función anterior, programe “`unsigned int unset1(unsigned int n)`” que cambia a 0 el bit más significativo de `n`.

Si `n == 0110101` entonces `n = 0010101`



Nañjeaqqóyvaoolución :P

```
unsigned int unset1(unsigned int n)
{
    unsigned int significativo = bit_mas_significativo(n);

    return ~significativo & n;
}
```



Pregunta 3

Programe la función “`unsigned char rotate_right(unsigned char n, unsigned char k)`” la cual realizará un shifteo y enviará el bit menos significativo de `n` y lo transformará en más significativo.

Si `n == 01011101` y `k == 2` entonces:

1. `10101110`

2. `01010111`



Solución

```
1 unsigned char rotate_right(unsigned char n, unsigned char k)
2 {
3     unsigned char bit;
4     unsigned int mask = 1;
5     int len = 8;
6     while(k)
7     {
8         k--;
9
10        bit = (n & mask) << (len - 1);
11        n = n >> 1;
12
13        n = (n | bit);
14    }
15    return n;
16 }
```



Pregunta 4

Programe la función “`void show_bits(int n)`” la cual recibe un número e imprime cada bit que contiene en la consola de forma ordenada.

Hint: Utilizar máscaras, la operación shift y putchar.



Solución

```
1 void show_bits(int n)
2 {
3     int bit_size = int_size(n);
4     unsigned int mask = 1;
5
6     for(int i = bit_size; i >= 0; i--)
7     {
8         int bit = (n >> i) & mask;
9         if(bit)
10             putchar('1');
11         else
12             putchar('0');
13     }
14     putchar('\n');
15 }
```



Last Question

Programar función “`int posicionBits(int bits, int subset bits, int size subset)`” la cual busca el patrón subset bits de largo size subset en bits.

- Retorna la posición si encuentra al patrón.
- Retorna -1 si no lo encuentra.



Solución

```
1
2
3
4
5
6
7
8 int posicionBits(int bits, int subset_bits, int size_subset)
9 {
10
11     int size = int_size(bits);
12     int mask = ~(~0 << size_subset);
13     int poscion = 0;
14
15     for (int i = 0; i <= size; i++)
16     {
17         if((bits & (mask << i)) == (subset_bits << i))
18         {
19             return i;
20         }
21     }
22
23     return -1;
24 }
25
26
27
28
29
30
31
32
33
```

