Programación de software de sistemas

Ayudantía 1

Profesor: Rodrigo Verschae

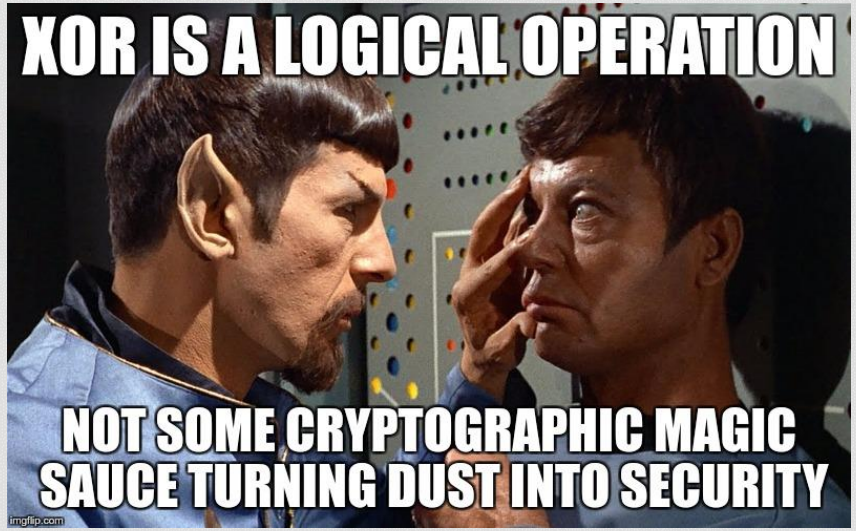
Ayudante: Nicolás Araya

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

```
int XOR(int n, int m)
{
     return (~n & m) | (n & ~m);
}

int XOR(int n, int m)
{
     return ~(n & m) & (n | m);
}
```

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

```
int int_size(int n)
{
        int c = 0;

        while(n)
        {
                n = n >> 1;
                c++;
        }

        return c;
}
```

```
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

Nah]easqdΦyvaolución :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

```
unsigned char rotate_right(unsigned char n, unsigned char k)
{
    unsigned char bit;
    unsigned int mask = 1;
    while(k)
    {
        k--;

        bit = (n & mask) << int_size(n);
        n = n >> 1;

        n = (n | bit);
    }
    return n;
}
```

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

```
void show_bits(int n)
{
        int bit_size = int_size(n);
        unsigned int mask = 1;

        for(int i = bit_size; i >= 0; i--)
        {
                int bit = (n >> i) & mask;
                if(bit)
                        putchar('1');
                else
                        putchar('0');
        }
        putchar('\n');
}
```

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

```
int posicionBits(int bits, int subset_bits, int size_subset)
{
    int size = int_size(bits);
    int mask = ~(~0 << size_subset);
    int poscion = 0;

    for (int i = 0; i <= size; i++)
    {
        if((bits & (mask << i)) == (subset_bits << i))
        {
            return i + sizeofBit(bits)-1; // De der a izq
        }
    }

    return -1;
}
```