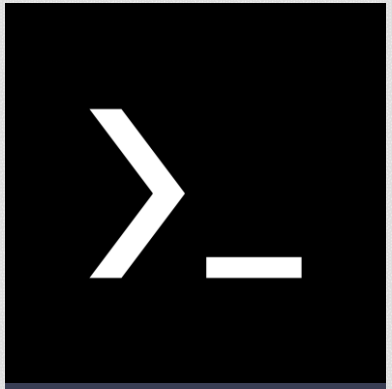
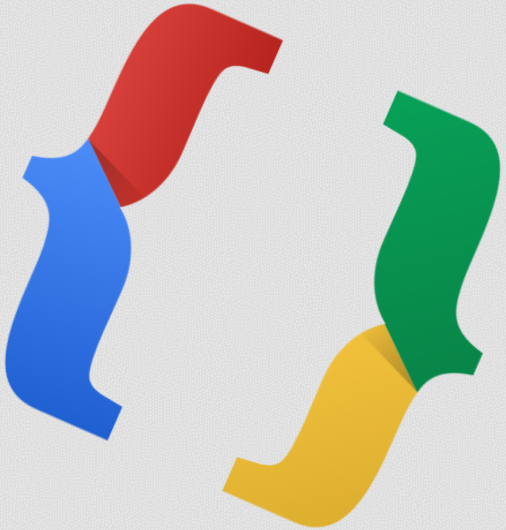
Programación de software de sistemas

Ayudantía 1

Profesor: Rodrigo Verschae

Ayudante: Nicolás Araya





Ubuntu

Tipos de variables

Tipo	Tamaño en bytes	Ejemplo
Int	2	1, 55, 73, 1500
Float	4	4.33, 5.92, 75.22, $5e^{-2}$
Double	8	7.5138, 9.513, $7e^{-5}$
Char	1	T, Y, %, i, #, 52
Void	0	No hay valor

Entrada y Salida

Salida:

Simbología	Resultado
%d	La variable tipo **entero** se muestra en **entero decimal**
%f	La variable **flotante** se muestra con **notación decimal**
%c	La variable **char** se muestra como **caracter**
%s	La variable se muestra como una **cadena de caracteres**
%o	La variable se muestra como **octal**
%x	La variable se muestra como **hexadecimal**
%u	La variable se muestra como un **entero sin signo**
%e	La variable se muestra como **notación científica**

Entrada y Salida

```c
#include <stdio.h>

int main()
{
	int c;
	while((c=getchar()) != EOF)
	{
		putchar(c);
	}
	return 0;
}
```

Otras formas de entrada y salida

```c
#include <stdio.h>

int main()
{
	int c;

	scanf("%d", &c);

	printf("%d", c);
}
```

Problema 1

Solicite al usuario, de manera amigable, ingresar una palabra e imprima en pantalla el código ASCII de cada letra y luego termine dándole las gracias.

Solución

```
#include <stdio.h>
int main()
{
        int c;

        printf(“Saludos usuario, ingrese una palabra por favor\n”);
        while((c=getchar())!= ‘\n‘)
        {
                printf(“%d,”, c);
        }
        printf(“\nGracias estimado usuario\n”);
        return 0;
}
```

Problema 2

Cree un programa que reciba un archivo con:

```
./programa.c < archivo.txt
```

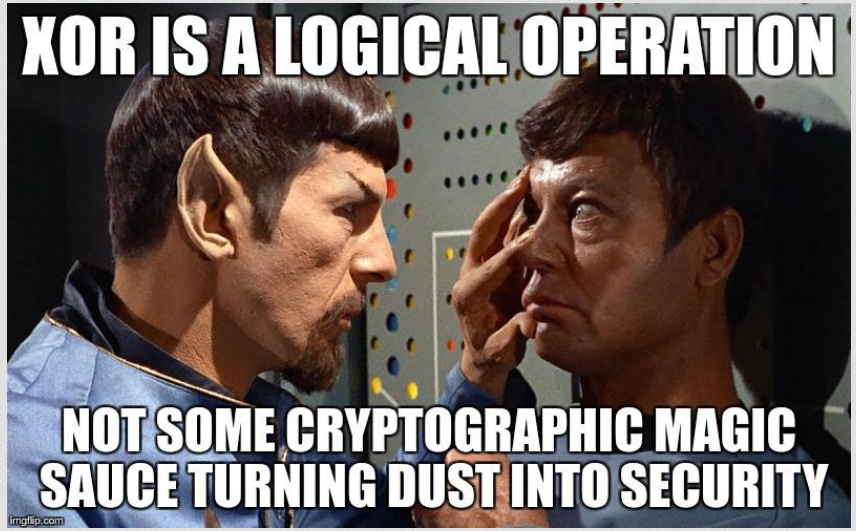
Que cuente cuántos char ‘a’ existen y que imprima en pantalla con printf el estado de la variable contador que usará.

Solución

```c
#include <stdio.h>

int main()
{
        int c;
        int a = 0;
        while((c=getchar()) != EOF)
        {
                if(c == ‘a’)
                {
                        a++;
                }
        }
        printf(“%d”,a);
}
```

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 3

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)?

Máscara de bits

Una máscara de bits corresponde a una cadena de bits utilizada para modificar otra cadena de bits de manera controlada mediante operaciones lógicas como AND, OR, XOR, NOT, etc.

Number => 10 => 000…0110

Mask => ~10 => 111…1001

Number & Mask => 0

Pregunta 4.1

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 4.2

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 5

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');
}
```