Linguagem C
Standard I/O

Operadores de bits

Ficheiros usando bibiloteca *standard* do C

Argumentos da função main

Operadores de bits

- O C têm operadores para manipular a representação binária dos números
  - Operadores lógicos:
    - `&` - e bit a bit
    - `|` - ou bit a bit
    - `^` - ou exclusivo bit a bit
    - `~` - negação bit a bit (complemento para 1)
  - Não confundir com os operadores lógicos booleanos - `&&`, `||` e `!`
  - Operadores de deslocamento de bits (shift)
    - `>>` - deslocamento de bits para a direita
    - `<<` - deslocamento de bits para a esquerda

Operadores lógicos bit a bit - revisão

e

a	b	a & b
0	0	0
0	1	0
1	0	0
1	1	1

ou

a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

ou exclusivo

a	b	a ^ b
0	0	0
0	1	1
1	0	1
1	1	0

negação

a	~a
0	1
1	0

Operadores lógicos bit a bit – uso

- Podemos usar os operadores lógicos para alterar certos bits de um número:
  - ✓ Por um bit a 0
  - ✓ Por um bit a 1
  - ✓ Inverter um bit
- Podemos usar **|** para por bits a **1**
  - ✓ **x ou 1** é sempre **1** (**1** - elemento absorvente do **ou**)
  - ✓ **x ou 0** é sempre **x** (**0** - elemento neutro do **ou**)
- Podemos usar **&** para por bits a **0**
  - ✓ **x e 0** é sempre **0** (**0** - elemento absorvente do **e**)
  - ✓ **x e 1** é sempre **x** (**1** - elemento neutro do **e**)
  - ✓ Às vezes é conveniente usar também a negação (**~**)

Exemplo ou: por um bit a 1

- Por o bit 3 de x a 1:

```
unsigned char x;
...
x = x | 8;                    // 8 = 2^3
```

X7	X6	X5	X4	X3	X2	X1	X0
				|			
0	0	0	0	1	0	0	0
				=			
X7|0	X6|0	X5|0	X4|0	X3|1	X2|0	X1|0	X0|0
				=			
X7	X6	X5	X4	1	X2	X1	X0

Exemplo and: por um bit a 0

- Por o bit 2 de x a 0:

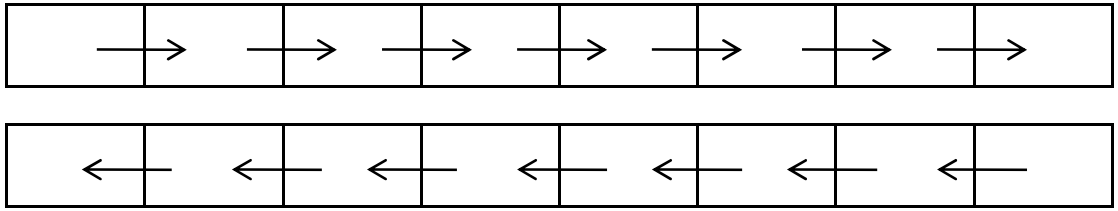
```
unsigned char x;

...

x = x & ~4;                    // 4 = 2^2
```

0	0	0	0	0	1	0	0
~							
1	1	1	1	1	0	1	1
&							
X7	X6	X5	X4	X3	X2	X1	X0
=							
X7&1	X6&1	X5&1	X4&1	X3&1	X2&0	X1&1	X0&1
=							
X7	X6	X5	X4	X3	0	X1	X0

Deslocamento de bits



- Os bits andam “n” casas para a direita ou para a esquerda
- Diferente para números com e sem sinal
  - ✓ Deslocamento aritmético vs deslocamento lógico
- Deslocar um número n bits para a esquerda corresponde a multiplicar por 2^n
- Deslocar um número n bits para a direita corresponde a dividir por 2^n

Deslocamento de bits - exemplo

```
unsigned char x = 25;

x = x << 1;      // x = x * 2
```

x

0	0	0	1	1	0	0	1

16 + 8 + 1 = 25

x << 1

0	0	1	1	0	0	1	0

32 + 16 + 2 = 50

Ficheiros em C

- Tipo *FILE* na biblioteca standard do C:
  - ✓ **`FILE *`**
- Ficheiros pré-definidos:
  - ✓ `FILE *`**`stdin;`** - entrada standard
  - ✓ `FILE *`**`stdout;`** - saída standard
  - ✓ `FILE *`**`stderr;`** - saída de erros standard
  - ✓ Normalmente associados à consola (terminal)
- Declarados em `<stdio.h>`
  - ✓ Como todas as funções para ficheiros.

Abrir e fechar ficheiros

- Abrir um ficheiro:
  - `FILE *fopen(char *pathname,char *mode);`
  - Mode pode ser:
    - **"r"** – leitura
    - **"w"** – escrita - cria ficheiro novo
    - **"r+"** – leitura/escrita
    - **"w+"** – leitura/escrita - cria ficheiro novo
    - Acrescenta-se **"b"** para ficheiros binários
  - Retorna **NULL** em caso de erro!
- Fechar um ficheiro:
  - **fclose**`(FILE* f);`
  - Garante que os dados são realmente escritos

Ficheiros de texto

- Leitura e escrita formatada:
  - **`fprintf`**`( FILE *f, char *fmt, ... );`
  - **`fscanf`**`( FILE *f, char *fmt, ... );`
  - `printf` e `scanf` para stdin e stdout
- Ler uma linha:
  - **`fgets`**`( char *linha, int max, FILE *f );`
    - Preenche "linha" com a próxima linha lida de f
    - Inclui o carácter `'\n'` na linha preenchida
    - Retorna NULL no fim do ficheiro
- Escrever uma linha
  - **`fputs`**`( char *linha, FILE *f );`

Cópia de ficheiros – linha a linha

```
#include <stdio.h>
#define MAXL 256

int main()
{  FILE *fin, *fout;
   char lin[MAXL];

   fin = fopen( "original.txt", "r" );
   fout = fopen( "copia.txt", "w" );
   if( fin == NULL || fout == NULL )
       // erro a abrir ficheiro
   while( fgets(lin, MAXL, fin) != NULL )
       fputs(lin, fout);
   fclose(fin); fclose(fout);
}
```

Leitura e escrita de caracteres (bytes)

- **`fputc`**`( int c, FILE *f );`
  - ✓ Escreve o carácter c no ficheiro f
- **`fputchar`**`( int c );`
  - ✓ Definido como `fputc( c, stdout )`
- `int` **`fgetc`**`( FILE *f );`
  - ✓ Le próximo carácter do ficheiro f
  - ✓ Retorna EOF no fim do ficheiro
  - ✓ EOF não é um caracter, mas um inteiro!
- `int` **`fgetchar`**`( void );`
  - ✓ Definido como `fgetc( stdin )`

Cópia de ficheiros - byte a byte

```
#include <stdio.h>

int main()
{  FILE *fin, *fout;
   int c;

   fin = fopen("original.txt", "rb" );
   fout = fopen("copia.txt", "wb" );
   if( fin == NULL || fout == NULL )
       // erro a abrir ficheiro
   while( ( c = fgetc(fin) ) != EOF )
       fputc(c, fout);
   fclose(fin);
   fclose(fout);
}
```

Argumentos da função main (1)

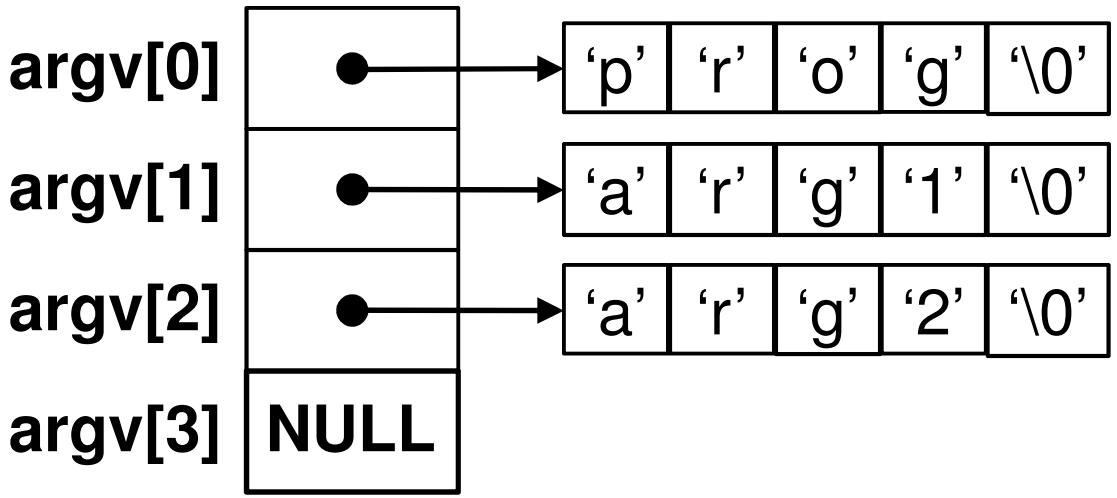
- A função main pode ter argumentos

```
int main( int argc, char *argv[] )
```

- argc – número de argumentos
- argv – vector de argumentos
  - ✓ Um vector de strings – apontadores para carácter!
- argv[0] corresponde ao nome do programa invocado

Argumentos da função main (2)

- Exemplo:

```
prog arg1 arg2
```

```
argc = 3
argv:
```