Função malloc

A função malloc (o nome é uma abreviatura de memory allocation) aloca um bloco de bytes consecutivos na memória do computador e devolve o endereço desse bloco. O número de bytes é especificado no argumento da função. No seguinte fragmento de código, malloc aloca 1 byte:

```
char *ptr;

ptr = malloc (1);

scanf ("%c", ptr);
```

O endereço devolvido por malloc é do tipo genérico void *. O programador armazena esse endereço num ponteiro de tipo apropriado. No exemplo acima, o endereço é armazenado num ponteiro-para-char.

Para alocar um tipo-de-dados que ocupa mais de 1 byte, é preciso recorrer ao operador sizeof, que diz quantos bytes o tipo especificado tem:

```
typedef struct {
    int dia, mes, ano;
} data;

data *d;

d = malloc (sizeof (data));

d->dia = 31; d->mes = 12; d->ano = 2014;
```

As aparências enganam: sizeof não é uma função mas um operador (tal como return, por exemplo).
A propósito, sizeof também pode ser aplicado a variáveis: se var é uma variável então sizeof var é o número de bytes ocupado por var. [Como var é uma variável, os parênteses na expressão sizeof (var) são redundantes.]

Overhead. Cada invocação de malloc aloca um bloco de bytes consecutivos maior que o solicitado: os bytes adicionais são usados para guardar informações administrativas sobre o bloco de bytes (essas

informações permitem que o bloco seja corretamente desalocado, mais tarde, pela função free). O número de bytes adicionais pode ser grande, e não depende do número de bytes solicitado no argumento de malloc. Não é ineficiente, portanto, alocar pequenos blocos de bytes; é preferível alocar um grande bloco e dele retirar pequenas porções na medida do necessário. Felizmente, malloc faz isso de maneira automática, sem que o programador/usuário perceba.