Introdução à Programação #8

Operadores

Operadores são símbolos para efetuar determinadas ações. Na programação existem diversos tipos de operadores. Operadores estes que também existem noutras áreas do conhecimento como matemática, por exemplo.

Serão abordados **3 tipos** de operadores: **aritméticos,** de **atribuição** e **relacionais**. O primeiro grupo de operadores serve, obviamente, para efetuar operações matemáticas. São utilizados os seguintes símbolos:

Operadores Aritméticos		
Nome	**Símbolo**	**Exemplo**
Soma	+	5 + 4 = 9
Subtração	-	154 – 10 = 144
Multiplicação	*	5,55 * 10 = 55,5
Divisão	/	40 / 2 = 20
Resto de uma divisão	%	1500 % 11 = 4

Os operadores de atribuição, tal como o próprio nome indica, servem para atribuir a uma variável, determinado valor. Existem vários operadores de atribuição e, muitos deles, funcionam como abreviatura para operações aritméticas.

Relembro que os operadores que aqui estão representados existem em muitas outras linguagens e que existem alguns outros dependendo da linguagem de programação.

Operadores de Atribuição	
Símbolo	**Exemplo***
=	var = 20 → **20**
+=	var += 5 *(**igual a** var = var + 5)* → **25**
-=	var -= 10 → **15**
***=**	var *= 4 → **60**
/=	var /= 5 → **12**
%=	var %= 5 → **2**

* Os exemplos aqui apresentados estão todos encadeados. O valor de *var* no segundo exemplo é atribuído no primeiro exemplo e assim consequentemente. Os valores a negritos precedidos por uma seta (→) correspondem ao valor com que a variável ficou.

Os operadores que agora vamos abordar, serão extremamente úteis para a matéria seguinte (*Loops*) pois permitem-nos efetuar comparações entre variáveis numéricas. Ora veja:

Operadores Relacionais		
Nome	**Símbolo**	**Exemplo**
Igualdade	==	x == y → **retorna TRUE se x for igual a y e FALSE se tiverem valores diferentes**
Diferente	!=	x != y → **retorna TRUE se x for diferente de y ou FALSE se x for igual a y**
Maior	>	x > 40
Maior ou Igual	>=	y >= 25
Menor	<	y < 20
Menor ou Igual	<=	x <= y

Loops "if else"

Os *loops "if else"* permitem executar determinados trechos de código dependendo do resultado de um teste lógico/condicional. A palavra *if* quer dizer "se" e *else* quer dizer, neste caso, "ou". A sintaxe deste *loop* é a seguinte:

```
1.  if (condição) {
2.      //Se a condição retornar TRUE então este código é executado.
3.  } else {
4.      //Caso contrário, isto é executado.
5.  }
```

Comentários

No código acima pode verificar que utilizámos duas barras (//) e depois escrevemos algo. Isto chamam-se comentários e são simplesmente ignorados pelo compilador.

Existem dois tipos de comentários: comentários de uma linha e comentários de várias linhas. Veja o seguinte exemplo:

```
1.  #include <stdio.h>
2.
3.  int main()
4.  {
5.      int n = 1; //Até ao final da linha é um comentário.
6.
7.      /*
8.       TUDO ISTO é um comentário
9.       */
10. }
```

Imagine que tem uma variável que contém um número qualquer e quer que o seu código imprima se a variável é maior ou igual a 50 ou menor que 50. Como vai proceder?

Para criar este algoritmos precisamos de inicializar uma variável, efetuar um teste condicional (utilizando um operador relacional) e depois executar o código em questão. Veja:

```
1.  #include <stdio.h>
2.
3.  int main()
4.  {
5.      int n = 25;
6.
7.      if (n >= 50) {
8.          printf("O valor %d é maior ou igual a 50.", n);
9.      } else {
10.         printf("O valor %d é menor que 50.", n);
11.     }
12.
13.     return 0;
14. }
```

Este trecho de código imprime uma mensagem a dizer que 25 é menor que 50. Como pode ver, isto apenas nos diz se a variável é menor que 50 ou então maior ou igual.

Mas, e seu quiser que as seguintes condições sejam testadas:

- Se for maior que 50 imprime X.
- Se for igual a 50 imprime Y.
- Se for menor que 50 imprime Z.

Podíamos efetuar este teste condicional da seguinte forma:

```
1.  if (n > 50) {
2.      printf("O valor %d é maior que 50.\n", n);
3.  }
4.
5.  if (n < 50) {
6.      printf("O valor %d é menor que 50.\n", n);
7.  }
8.
9.  if (n == 50) {
10.     printf("A variável é igual a 50.\n");
11. }
```

Isto tornaria o código um pouco repetitivo. Podemos utilizar o comando *else if* que quer dizer "ou se" da seguinte forma:

```
1.  if (n == 50) {
2.      printf("A variável é igual a 50.\n");
3.  } else if (n < 50) {
4.      printf("O valor %d é menor que 50.\n", n);
5.  } else {
6.      printf("O valor %d é maior que 50.\n", n);
7.  }
```

Como gerar um número aleatório

Nos exercícios seguintes, irá precisar de gerar um número aleatório. Para o fazer, irá necessitar de incluir mais um ficheiro *.h* que se chama *time.h*.

Pode utilizar a função para gerar números aleatórios da seguinte forma:

```c
#include <stdio.h>
#include <time.h>

int main()
{
    //Inicializar a seed para o número aleatório
    srand (time(NULL));
    //Atribuir o número aleatório (entre 1 e 10) a uma variável
    int n = (rand() % 10) + 1;

    return 0;
}
```

Mais tarde falaremos melhor sobre esta função.

Artigo original: http://pplware.sapo.pt/tutoriais/vamos-programar-introducao-a-programacao-8