DEEC
DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Programação de Computadores (2013/14)

Primeira folha de Problemas

Matéria abordada: Primeiro contacto com o ambiente de trabalho. Escrita de mensagens no ecrã. Expressões numéricas. Atribuição
Bibliografia: Capítulo 3, páginas 23-34 de [Oua03]. e Capítulo 4, páginas 35-47 de [Oua03].

Em primeiro lugar leia o documento "Ambiente de Trabalho".
Depois de fazer *login* abra uma consola (para ter acesso à linha de comandos).

---

1. No InforEstudante vá ao material e apoio o obtenha o ficheiro CodigoFonte.zip. Obtenha o ficheiro f1e1.cpp ou utilize o *geany* para escrever o seguinte programa fonte (adaptado de [Oua03, pág 25]).

```
//=================================================
// O meu primeiro programa em c++
//
// Autor:  ...                          Data:2013/mm/dd
//
// Objectivo: Demonstracao de um programa simples (1)
//
// Utilizacao: Execute que a mensagem aparece
//=================================================

#include <iostream>

int main()
{
   // Escreve Bom dia! e muda de linha
    std::cout << "Bom dia!\n";
    return(0);
}
```

   Designe esse ficheiro por `f1e1.cpp`. Salve-o na directoria `~/Desktop/aula1` (que pode criar na altura em que gravar o ficheiro).

   (a) Abra uma linha de comando e compile o ficheiro:

   ```
   g++ -c f1e1.cpp
   ```

   Verifique que foi criado o ficheiro de nome `f1e1.o` (usando o gestor de ficheiros `dolphin` ou simplesmente fazendo `ls -la` (ou `dir`) na linha de comando).

   (b) Crie o ficheiro executável, a partir de `f1e1.o`, fazendo:

   ```
   g++ -o f1e1.exe f1e1.o
   ```

(c) Execute o ficheiro `f1e1.exe` escrevendo na linha de comando (assumindo que está em `~Desktop/aula1`):

`./f1e1.exe` <*return*>

Nota 1: O `<<` é um operador e pode ser designado de insersor sobre `cout`.

2. O objecto `cout` permite-nos apresentar informação no terminal em que o programa está a executar. Este objecto pertence ao *namespace* `std`, na qual o C++ define a sua *standard library* – a definição de um *namespace* será aflorado mais adiante na disciplina.

   Para não precisarmos de estar sempre a escrever `std::cout`, podemos escrever o programa anterior da seguinte forma:

```
//=================================================
// O meu segundo programa em c++
//
// Autor:  ...                          Data:2013/mm/dd
//
// Objectivo: Demonstracao de um programa simples (2)
//
// Utilizacao: Execute que a mensagem aparece
//=================================================

#include <iostream>

using namespace std; // para nao ser preciso std::

int main()
{
   // Escreve Bom dia! e muda de linha
    cout << "Bom dia!\n";
    return(0);
}
```

   A frase "`using namespace std;`" indica que, por omissão, objectos não declarados deverão pertencer ao *namespace* `std`.

   (a) Copie o ficheiro anterior (`f1e1.cpp`) para `f1e2.cpp` e modifique-o de forma a que o seu conteúdo coincida com o exemplo dado.
   Para copiar o ficheiro use o comando `cp f1e1.cpp f1e2.cpp` ou utilize o *dolphin*.

   (b) Combine agora os passos de compilação e ligação:

```
g++ -o f1e2.exe f1e2.cpp
```

   Verifique que não foi criado o ficheiro `f1e2.o`, mas apenas o ficheiro `f1e2.exe`.

   (c) Execute o ficheiro `f1e2.exe`.

Nota 2: Sempre que o seu programa se encontrar num único ficheiro é mais prático utilizar a opção "`-o`" do que a opção "`-c`", pois a primeira cria logo um executável – **e será essa a opção mais usual nas aulas de Programação de Computadores**. Em projectos com

alguma dimensão, que se estendem por vários ficheiros, é útil/necessário poder compilar cada um em separado, criando os respectivos ficheiros de extensão "`.o`", antes de criar o executável.

3. Copie o ficheiro `f1e2.cpp` para `f1e3.cpp`. Acrescente a linha:
`cout << "Boa tarde, ou boa noite..." << endl;`

```
//=================================================
// O meu terceiro programa em c++
//
// Autor: ...                          Data:2013/mm/dd
//
// Objectivo: Demonstracao de um programa simples (3)
//
// Utilizacao: Execute que as mensagens aparecem
//=================================================

#include <iostream>

using namespace std;  // para nao ser preciso std::

int main()
{
    // Cumprimentando
    cout << "Bom dia!\n";
    cout << "Boa tarde, ou boa noite..." << endl;
    return(0);
}
```

Nos problemas anteriores (e neste também!) conseguiu uma mudança de linha utilizando o caracter `'\n'`. Utilizando `endl` consegue-se também um fim de linha; mas `endl` produz um fim de linha e força o esvaziamento do *buffer* de saída (o que não acontece quando utilizamos o caracter `'\n'`).

A diferença entre a utilização do caracter `'\n'` *e de* `endl` *não é fácil de observar, mas existe!*

Crie o ficheiro executável correspondente ao ficheiro fonte `f1e3.cpp`, designe-o por `f1e3.exe` e execute-o.

Nota 4: Tal como anteriormente, escrevemos `using namespace std;` antes da função `main()`, tornando desnecessário preceder `cout` e `endl` de `std::`

```
cout << "Boa tarde, ou boa noite..." << endl;
```

Nota 5: Caracteres reservados ou não visíveis são obtidos usando uma `\` antes do código ou símbolo correspondente. Por exemplo para apresentar no ecrã
`Barra: \`
terá de no seu programa incluir a seguinte linha:
`cout << "Barra: \\" << endl;`
Questão: para obter `"` (o carácter reservado que indica o fim de uma sequência de caracteres) no ecrã, como deverá fazer?

4. Em C++ podemos calcular o valor de expressões numéricas, utilizando os símbolos +, -, *, /, para representar a adição, subtracção, muliplicação e divisão, respectivamente. Podemos igualmente usar parêntesis curvos para agrupar elementos.

   O C++ considera que números como 5 ou 3 são números inteiros, e ao dividir 5 por 3, executa uma divisão inteira, desprezando o resto: o quociente de 5/3 vale 1 (e o seu resto é 2).

   (a) Qual será o resultado do seguinte programa?
   Responda a esta questão num ficheiro de texto f1e4.txt, **sem transcrever, compilar e executar o programa**!

```
//================================================
// Calculando o valor de expressões aritméticas
// Autor: ...                          Data:2013/mm/dd
// Objectivo:  Calculando e mostrando
//             o valor de expressões aritméticas
// Utilização: Execute que as mensagens aparecem
//================================================
#include <iostream>
using namespace std;

int i, j, k; // Variáveis usadas nos exemplos

int main()
{
   // Expressões com constantes
   cout << "\n5 * 3 = "  <<  5 * 3 << endl;

   // Instrução inútil
   (3 + 1) * 8;  // calcula 32 e passa à instrução seguint

   // Expressões com variáveis
   i = 7;  // i <-- 7
   j = 3;  // j <-- 3
   // Calcula e mostra, mas não armazena!
   cout << "\ni * j = "  <<  i * j;
   // i e j ficaram inalterados
   cout << "\ni = " << i <<  " e  j = " << j<< endl;

   // Atribuicoes
   k = i - j; // Calcula e guarda o resultado em k (k <-- i - j)
   // i e j continuam inalterados
   cout << "\nDepois de k = i - j, i = " << i <<  " e  j = " << j;
   cout << "\nk = "  <<  k;
   k = k + 1; // k <-- k + 1
   // Mostra k após incremento de um
   cout << "\nDepois de incrementar k = "  <<  k << endl;
   return(0);
}
```

(b) Obtenha este programa de `CodigoFonte.zip`. Se isso não for possível, escreva este programa usando o *geany*. Designe o ficheiro correspondente por `f1e4.cpp`.

Compile e execute-o. Verifique se as saídas apresentadas coincidem com o que esperava encontrar.

5. Em C++ as operações entre inteiros têm um resultado do tipo inteiro. Os exercícios seguintes procuram ajudar a visualizar a diferença entre as operações realizadas entre inteiros e entre números em virgula flutuante.

(a) Complete a tabela, sabendo que i e j são duas variáveis inteiras do tipo `int`.

`int` i	`int` j	`int` $i*j$	`int` i/j
12	4		
12	5		
3	6		

(b) Escreva um programa (f1e5a.cpp) em que declara duas variáveis inteiras `int i, j;` Solicita dois números inteiros ao utilizador e em seguida mostra ao utilizador o seu produto e o seu quociente:

```
cout << "\nMultiplicando: " << i << '*' << j << '=' << i*j ;
cout << "\n    Dividindo: " << i << '/' << j << '=' << i/j ;
```

Verifique que preencheu a tabela correctamente.

(c) Complete a tabela, sabendo que x e a y são duas variáveis inteiras do tipo `float`.

`float` x	`float` y	`float` $x*y$	`float` x/y
12	4		
12	5		
3	6		

(d) Escreva um programa (f1e5b.cpp) em que declara duas variáveis capazes de armazenar valores reais `float x, y;`

E repita alínea 5b, verificando que preencheu correctamente a tabela anterior.

Note que, se fizer:
`cout << "3.0 == " << 3.0;`
no ecrã surge: `3.0 == 3`, porque por omissão só são apresentadas as casas decimais de um **número** se estas não forem zero.

TPC Escreva (usando o *geany*.) um programa em C++ que lhe permita visualizar a diferença entre as operações realizadas entre inteiros e entre números em virgula flutuante.

Escreva (usando o *geany*.) um programa em C++ que solicita ao utilizador dois número inteiros e em seguida apresenta ao resultado das quatro operações elementares (adição, subtracção, multiplicação e divisão) em mensagens apropriadas (siga o estilo do exercício 5b).

Apresente também o resto da divisão do primeiro número dado pelo segundo introduzido.

Referências

[Oua03] S. Oualline. *Practical C++ Programming*. O'Reilly, 3rd edition, 2003.