DEEC
DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Programação de Computadores (2013-2014)

Sexta folha de Problemas

Matéria abordada: funções (passagem de parâmetros por valor), alcance das variáveis, ciclos (while, do while, for), switch; gama dos números num computador.
Bibliografia: Capítulo 5, 6 e 8 de [Oua03].
De agora em diante estão proibidas as variáveis globais!

1. Reutilize o código que implementou na resolução do problema 2 da folha 5 para resolver o seguinte problema:

   (a) Projecte e implemente uma função que dado um carácter e um inteiro (número ímpar compreendido entre 5 e 21, inclusive) desenha um '0', tal como é exemplificado para o carácter '*' e o número 5.

   Desenhe a letra considerando uma grelha quadrada (obviamente não precisará de desenhar a grelha). Exemplo, quando o utilizador dá o valor 5:

	*	*	*	
*				*
*				*
*				*
	*	*	*	

   O prótótipo da função é `void desenhaZero(char elem, int dim);`

   Verifique a correcção sintáctica da função compilando o ficheiro onde a definiu:

   ```
   g++ -c f6e1.cpp
   ```

   (b) Escreva um programa (completando o ficheiro anterior), que utilizando a função "`desenhaZero()`", desenha um '0', utilizando o símbolo e as dimensões dadas por um utilizador. Esse programa deve solicitar ao utilizador o carácter a usar e as dimensões do '0', a desenhar, tal que o número dado seja um número ímpar compreendido entre 5 e 21 (inclusive).

2. Neste exercício pretende-se fazer uma função que retorne o resultado de uma operação aritmética, funcionando como uma calculadora simples.
   Irá precisar de fazer `#include <cfloat>` para incluir as definições de constantes com os valores limite dos tipos em vírgula flutuante no seu sistema e compilador. Nesse ficheiro estão definidas (entre outras constantes) `FLT_MAX` e `FLT_MIN`. Pode fazer:
   ```
   cout << "\nFLT_MAX = " << FLT_MAX;
   cout << "\nFLT_MIN= " << FLT_MIN;
   ```
   para ver qual o valor destas constantes no sistema e compilador que está a utilizar.

   (a) Considere o seguinte protótipo:
   ```
   float calculadora(char operacao, float a, float b);
   ```
   para uma calculadora simples. Implemente a função `caluladora` utilizando um `switch` tendo em conta a tabela seguinte.

Operação	Operação aritmética	Valor retornado
'+'	Adição	a+b
'-'	Subtracção	a-b
'*'	Multiplicação	a*b
'/'	Divisão	a/b se b $\neq 0$ e no caso de $b = 0$ vale `FLT_MAX` se $a > 0$ ou `FLT_MIN` se $a < 0$ ou 0 se $a = 0$.
Outro carácter		0

Tabela 1: Descrição do funcionamento da calculadora.

   Verifique a correcção sintáctica da função compilando o ficheiro onde a definiu:
   `g++ -c f6e2.cpp`.

   (b) Escreva um programa que solicite ao utilizador um carácter (que indique a operação a realizar) e um par de números reais (operandos). De seguida deve apresentar o resultado da operação sobre esse par de números, utilizando a função `caluladora()`.
   O programa deve executar a tarefa anterior **repetidamente**, até que seja dado o carácter '.' para terminar.
   O seu programa não deve aceitar para `operacao` nenhum carácter além de +, -, *, e / ou . (para terminar).

3. Dado um número entre 2 e 9 (que deverá ser pedido ao utilizador), escreva uma função que desenhe no monitor um triângulo rectângulo em que ambos os catetos têm a dimensão dada pelo número introduzido.

   Comece por definir o protótipo das funções. Na alínea 3a) designe a função por `trianguloX` e na na alínea 3b) designe a função por `trianguloN`.

   Se, por exemplo, for dado o número 5 obteríamos:

   (a)
   ```
   x
   xx
   x x
   x  x
   xxxxx
   ```

   (b)
   ```
   1
   22
   3 3
   4  4
   55555
   ```

   (c) Escreva um programa principal que garanta que não é aceite nenhum número fora do intervalo [2,9] e em seguida desenha sucessivamente cada um dos triângulos (deixe pelo menos um linha de intervalo entre eles), utilizando as funções implementadas.

Nota: Antes de começar a escrever as funções, defina uma estratégia de resolução (que deverá ser comum a ambas as funções).

4. (a) Escreva o protótipo de uma função que dados os dois catetos de um triângulo rectângulo devolve a sua hipotenusa.

   (b) Implemente a função cujo protótipo escreveu na alínea anterior.

   Irá precisar de fazer "`#include <cmath>`", para usar a função `sqrt()`, que calcula a raiz quadrada.

   Verifique a correcção sintáctica da função compilando o ficheiro onde definiu a função.

   (c) Implemente uma função, com o seguinte protótipo:

   ```
   float lefloat();
   ```

   que peça um número `float` estritamente positivo, e só termina quando o utilizador dá um número estritamente positivo.

   Sugestão: na função utilize um ciclo `do while`.

   Verifique a correcção sintáctica da função compilando o ficheiro onde a definiu.

   (d) Escreva um programa principal que pede dois números reais estritamente positivos (a medida dos catetos) e mostra a medida da hipotenusa, utilizando as funções anteriores.

5. Escreva o seguinte conjunto de funções e no final escreva um programa que lhe permita testar o seu funcionamento:

   (a) `bool edigito (char a);` – dado um carácter `a` diz se é um dígito ('0' a '9').

   (b) `bool eletra (char a);` – dado um carácter `a` diz se é uma letra ('a' a 'z' ou de 'A' a 'Z').

   (c) `bool emaiuscula (char a);` – dado um carácter `a` diz se é uma letra é maiúscula ('A' a 'Z').

   (d) `bool eminuscula (char a);` – dado um carácter `a` diz se é uma letra é minúscula ('a' a 'z').

   (e) `int letra2int (char a);` – dado um carácter `a` que é um dígito ('0' a '9') devolve o inteiro correspondente. Se `a` valer '3' então o resultado deve ser 3 (inteiro).

   Se o carácter dado não for um inteiro, devolve -1.

   Utilize a função `bool edigito (char a);` para saber se o carácter dado é um dígito.

6. Aceda a uma página, como por exemplo:
   `http://www.cppreference.com/`
   ou
   `http://www.cplusplus.com/`
   e identifique nas bibliotecas sobre manipulação de caracteres (*Character handling functions*, `cctype`) funções semelhantes às que acabou de implementar no problema anterior.

7. Considere o programa:

```
//=================================================
// Apresenta os multiplos de 3 \in [1.. n], n > 3
//
// Autor: Nome Apelido               Data:2011/mm/dd
//
// Objectivo: estudar o alcance/ambito das variaveis
//            incluindo um ciclo do while
//=================================================
#include <iostream>

using namespace std;

int n = 2; // n e' uma variavel global

int main()
{
    do {
        // Nova variavel muito local. Torna o 'n' global invisivel.
        int n = 0;
        cout << "De um inteiro maior do que 3: ";
        cin >> n;
        if (n <= 3) cout << "Valor Invalido. Tente de novo." << endl;
    } while ( n <= 3 ); // Este n e' o muito local ou o global?

    cout << "\nVou mostrar os multiplos de 3 <= " << n << endl;

    int i; // Contador (variavel) local
    for (i=1; 3*i <= n; ++i)
        cout << 3*i << endl;

    return(0);
}
```

O programa anterior compila sem erros, *mas tem problemas.*

(a) Uma vez iniciada sua execução, este programa fica em ciclo eterno. Porquê?

(b) Que sugere como solução para corrigir esse ciclo? A solução é única? Se não for, indique outra.

8. Considere o programa:

```
//=======================================================
// Mostrar os divisores de um numero
//
// Autor: Nome Apelido               Data:2011/mm/dd
//
// Objectivo: estudar o alcance de variaveis
//            declaradas dentro de ciclos
//=======================================================
#include <iostream>

using namespace std;

int main()
{

    do {
        cout << "De um inteiro positivo: ";
        int n;
        cin >> n;
        if (n < 0)
            cout << "Valor negativo.  Tente de novo." << endl;
        else if (n == 0)
            cout << "Valor nulo.  Tente de novo." << endl;
    } while (n <= 0);

    for (int div = 0; div <= n; ++div)
        if ( n % div == 0 )
            cout << '\n' << div << "e' divisor de " << n << endl;

    cout << "\nEste ciclo termina quando " << div
         << " e' igual a " << n+1;
    return(0);
}
```

(a) O programa anterior não compila. Porquê?

(b) Corrija-o de forma a que o compilador crie o ficheiro executável correspondente.

(c) Execute-o. Analise o resultado e corrija novamente o programa se tal for necessário. Se tiver sido necessário mais uma correcção repita a sua execução, analise novamente os resultados e verifique que detectou todos os problemas e que os corrigiu devidamente.

9. Considere o programa:

```
//=============================================================
// Calcula e mostra min(x,y)
//
// Autor: Nome Apelido               Data:2011/mm/dd
//
// Objectivo: estudar using e namespaces
//=============================================================
#include <iostream>

using namespace std;

float min; // A var min é global

int main()
{
    float x, y;

    // Le dois numeros
    cout << "\nDois numeros: ";
    cin >> x >> y;

    //O Calcula o min
    min = ( x < y ? x : y);

    // Mostra o min(x,y)
    cout << "\nmin( " << x << ", " << y << ") =  " << min << endl;

    return(0);
}
```

(a) Tente compilar o programa. Leia com atenção a primeira mensagem de erro que surge na tentativa de compilação.

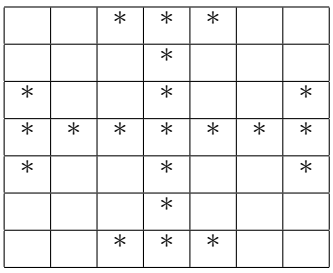
Qual a origem do erro?

(b) Existem várias formas de resolver este problema, sem alterar o nome de nenhuma variável presente no programa dado.

Escolha duas delas. Teste-as.

10. (a) Projecte e implemente uma função que desenhe no ecrã uma cruz de cristo simplificada com asteriscos.

    Desenhe a cruz considerando uma grelha quadrada (obviamente não precisará de desenhar a grelha). Exemplo, quando o utilizador dá o valor 7:



    O número de estrelas no topo de cada braço da cruz deve ser o ímpar mais próximo de metade da medida do lado. Exemplos:
    `5.0/2.0 = 2.5`, logo deverá haver 3 estrelas no topo do braço
    `7.0/2.0 = 3.5`, logo deverá haver 3 estrelas no topo do braço
    `9.0/2.0 = 4.5`, logo deverá haver 5 estrelas no topo do braço

    Existem quatro tipos de linha e utilizará quatro funções, uma para desenhar cada uma delas (mudando de linha no final):

    i. As linhas no topo e na base: `void topo_ou_base(int lado);`
    ii. A linha do meio (horizontal): `void meio(int lado);`
    iii. As linhas que apenas desenham o eixo vertical central: `void eixo(int lado);` (ou seja as linhas entre a linha de topo e a primeira estrela dos braços laterais e as linhas entre a linha com a última estrela dos braços laterais e a linha da base);
    iv. As linhas antes e depois da linha do meio, que (juntamente com a linha do meio) constroem os braços lateriais: `void bracos(int lado);`

    Implemente estas quatro funções. Escreva a função `void cruz(int dim)` chamando as quatro funções cujo protótipo lhe foi dado.

    Na resolução desta alínea utilize pelo menos um ciclo `while` e um ciclo `for` (ou seja não resolva o problema utilizando apenas ciclos `while` ou apenas ciclos `for`).

    (b) Escreva um programa principal que solicita a dimensão do lado da cruz, a qual deve ser um número ímpar compreendido entre 5 e 21 (inclusive). Garanta que o utilizador dá um número ímpar no intervalo de valores requerido para a dimensão do lado da cruz.

    Sugestão: Na leitura da dimensão da grelha em que desenhará a cruz utilize um `do while`.

11. (a) Projecte e implemente uma função, `ziguezague`, que desenhe no ecrã um friso vertical de largura n e altura $n * k$. Os frisos têm dois elementos que se sucedem alternadamente: diagonal da esquerda para a direita e da direita para a esquerda (ver exemplo). Cada elemento no friso utiliza duas funções:

    i. A diagonal da esquerda para a direita: `void esquerda_para_direita(int lado, char c);`

    ii. A diagonal da direita para a esquerda: `void direita_para_esquerda(int lado, char c);`

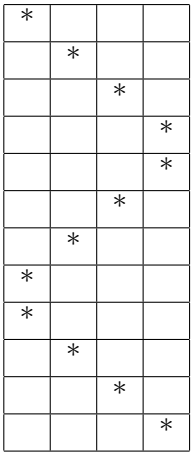
    Implemente estas duas funções.

    Escreva a função de protótipo:

    ```
    void ziguezague(int n, int k, char c);
    ```

    utilizando as duas funções anteriores e começando sempre com a função `esquerda_para_direita()`.

    Exemplo quando o utilizador dá o valor $n = 4$ para a largura, $k = 3$ para o número de elementos do friso em ziguezague, e $c =$'*' para o símbolo utilizado na sua construção:



    A grelha quadrada é auxiliar para a visualização e, obviamente, não precisa de a desenhar.

    (b) Escreva um programa principal que lhe permita testar a função `friso` e que apenas permita: $n \in [3, 9]$, $k \in [1, 10]$, $c \in$ {'*','+','#','x'}.

TPC Dado um número entre 2 e 40 (que deverá ser pedido ao utilizador), escreva uma função que desenhe no monitor um triângulo rectângulo em que ambos os catetos têm a dimensão dada pelo número introduzido.

Comece por definir o protótipo da função (designe a função por `trianguloX2`).

Se, por exemplo, for dado o número 5 obteríamos:

(a)

```
    x
   xx
  x x
 x  x
xxxxx
```

(b) Escreva um programa principal que garanta que não é aceite nenhum número fora do intervalo [2,40] e em seguida desenha um triângulo utilizando a função implementada.

Referências

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