Universidade do Minho	Escola de Engenharia da Universidade do Minho **Mestrado Integrado em Eng. Electrónica Industrial e Computadores** # Complementos de Programação de Computadores	**2014/2015** **MIEEIC** **(1º Ano)** **2º Sem**
Aula Prática 1: Exercícios de Revisão de C e Introdução ao C++		

Objectivos:

Esta Folha de Exercícios destina-se a:

- Recordar ou apresentar conceitos rudimentares de programação prática.
- Realizar uma revisão de conceitos de programação em linguagem C;
- Facilitar a transição do ambiente de programação da linguagem C para o da linguagem C++;

Os exercícios aqui propostos deverão ser realizados no mais simples ambiente de desenvolvimento possível para a linguagem C: DevC++, Codeblocks, editor de texto de programação, ferramentas da GCC (GNU Compiler Collection) e/ou afins.

Exercício 0 – Revisões de Linguagem C

a) Escreva uma função `int troco(int vNota, int aPagar)` que dado um determinado valor inteiro a pagar igual ou inferior a 50 Eur (`aPagar`) e o valor da nota entregue para o pagar (`vNota`) que pode ser (50, 20, 10 ou 5 Eur) calcule o troco a entregar (exclusivamente em notas de 20 e 5 Eur e moedas de 1 Eur) e o imprima no écran. A função deve ainda imprimir e retornar o número de notas/moedas necessário para o troco. No caso de a nota entregue ou o valor a pagar serem inválidos deve imprimir inválido e retornar 0. Exemplos:

```
troco(38,15) -> Inválido!
troco(50,25) -> 20 5: 2 notas/moedas
troco(50,20)-> 20 5 5: 3 notas/moedas
troco(20,4)-> 5 5 5 1: 4 notas/moedas
troco(5,1) -> 1 1 1 1: 4 notas/moedas
```

b) Construa a função `void num7(int dim)` que dada a dimensão de um número sete `dim` desenhe números 7 compostos por caracteres 'X' com aspeto semelhante aos exemplos apresentados abaixo:

```
num7(1)     num7(3)     num7(4)      num7(5)
X           XXX         XXXX         XXXXX
X             X           X             X
num7(2)      X           X            X
XX          X           X            X
 X                      X           X
X                                   X
```

Suponha que dispõe da seguinte função já implementada:

```
void nch(int n, char c) {
    for(int i=0; i<n; i++) printf("%c", c); }
```

c) Escreva uma função `double frequenciasPal(char nome[20])` que receba uma string indicando o nome de um ficheiro e imprima no écran uma tabela indicando a frequência de cada comprimento das palavra. A função deve também escrever no écran o comprimento médio das palavras encontradas no ficheiro e retorna-lo. Suponha que o tamanho máximo das palavras é 10. Sugestão utilize um vetor de inteiros para as frequências: `int freq[10]` inicializado a 0.

Exemplo:

```
1 ->  50
2 -> 180
3 -> 220
…
10 ->  8
Med: 4.6
```

d) Suponha a estrutura `tempo` seguinte:

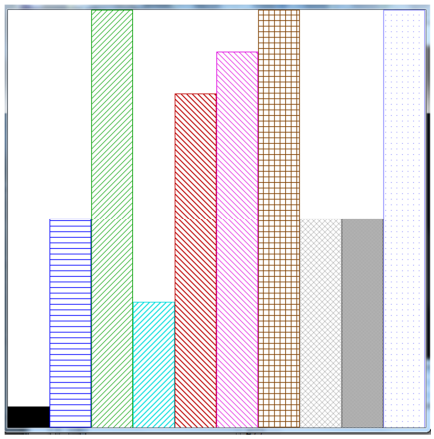
```
struct tempo { short int hora, minu, segu; };
```

Escreva uma função `void incr100(tempo &t)` que dado um determinado tempo, o incremente em 100 segundos, alterando convenientemente os respetivos campos e o mostre no ecrã devidamente formatado (hora, minutos e segundos). Por exemplo, supondo `t = {10, 59, 12}` a chamada de `incr100(t)` resulta em t=`{11, 00, 52}` e imprime no écran "11 horas, 00 minutos e 50 segundos".

e) Suponha uma matriz, já inicializada `m[N][N]` contendo números inteiros representando o estado de um jogo com peças de seis tipos 1-6, e ainda armadilhas (8) e espaços vazios (0) num tabuleiro com dimensão `NxN`. Construa a função `int move(int m[N][N])` que dada a matriz com o estado do jogo execute a movimentação do jogo que consiste nos seguintes passos. A) As armadilhas que não tenham peças à sua volta (só espaços vazio ou outras armadilhas) no início do ciclo, expandem-se (caso seja possível) para baixo embora as novas armadilhas não capturem peças neste ciclo, só no próximo. b) Todas as peças com valor superior capturam as peças de valor menor que estejam a cima, baixo, esquerda e/ou direita da peça. c) As quadrículas que estejam próximas de armadilhas ficam vazias mas só após a expansão feita em b) caso seja o caso. A função deve retornar o número de peças alteradas no tabuleiro.

Matriz m0:	move(m0)	move(m0)	move(m0)
1 1 0 2 4	1 **4** 0 **4** 4	**4 6** 0 4 4	**6** 6 0 4 4
1 4 0 1 0	**4 6** 0 **2** 0	**6** 6 0 **4** 0	6 6 0 4 0
0 6 8 2 0	0 **0** 8 **0** 0	0 0 8 0 0	0 0 8 0 0
1 6 2 6 3	**0** 6 **0** 6 **6**	0 6 **8** 6 6	0 **0** 8 **0** 6
8 2 3 2 4	8 **0** 3 **6** 4	8 0 **6** 6 **6**	8 0 **0** 6 6
Valor de Retorno:	12	7	4

f) construa uma função `void visualiza (int vec[10])` que permita visualizar em modo gráfico (utilizando a interface BGI descrita nas aulas e acetatos da disciplina) o conteúdo de um dado vetor de números inteiros contendo 10 valores de percentagens (valores entre 0 e 100). A visualização deve ser numa janela de fundo branco, aproximadamente 600x600 pixéis, através de barras coloridas, representadas verticalmente, com uma cor distinta e um padrão de enchimento distinto. Exemplo para vetor contendo vec[10]={5, 50, 100, 30, 80, 90, 100, 50, 50, 100};



g) Construa um programa principal que lhe permita testar devidamente todos os exercícios anteriores (11-16).

Exercício 1

Pretende-se que sejam escritas algumas variantes do conhecido "1º programa" de qualquer linguagem de programação: "Hello, World!". O aspecto comum a todas as alíneas é que cada programa escreva no ecrã a famosa frase um certo, e específico, número de vezes. Os programas deverão ser escritos em C++.

a) Escreva o programa base "Hello World!".

b) Escreva o programa pedido, de forma a que o número de vezes que a frase deve surgir no ecrã seja indicado na linha de comando.

c) Escreva o programa pedido, mas agora de forma a que o número de vezes que a frase deve surgir no ecrã seja recolhido da consola.

d) Escreva o programa pedido mas, desta vez, uma frase qualquer, recolhida da consola, é que é mostrada no ecrã tantas vezes quantas o número indicado na linha de comando.

Exercício 2

Pretende-se escrever um programa em C que calcule a hipotenusa de um triângulo rectângulo. Para tal efeito, os valores das dimensões dos catetos deverão ser obtidos do utilizador de diferentes maneiras.

a) Obtenha as dimensões dos catetos da linha de comando, e apresente na consola, o valor da hipotenusa (note que as medidas de qualquer um dos lados do triângulo poderão ser valores em vírgula flutuante).

b) Continuamente, obtenha da consola as dimensões dos catetos (sempre positivos) e produza o valor da hipotenusa até que o utilizador escreva um valor negativo.

Exercício 3

Relativamente a números primos:

a) Escreva uma função que identifique números primos (e insira-a num programa de teste em C). O protótipo da função é: int primo(int n); e retorna 1 se n é primo e 0 no caso contrário. (Sugestão: para verificar se n é primo, basta dividir n pelos números de 2 a sqrt(n).)

b) Recorrendo à função anterior, escreva um programa que imprima os números primos compreendidos entre 1 e N, em que N é um valor indicado pelo utilizador na linha de comando.

c) Escreva um programa em que na linha de comando é especificado como argumento o nome de um ficheiro onde se encontram os números que se quer determinar se são primos ou não. O programa deve imprimir na consola para cada número lido uma linha com (o parenteses reto denota a alternativa): <número>: primo [ou: não primo]

d) Escreva um programa idêntico ao da alínea anterior, mas em que na linha de comando é especificado como 2º argumento o nome de um ficheiro onde se deve colocar as linhas antes impressas na consola.

Exercício 4

a) Escreva um programa que solicite ao utilizador um número e que depois imprima todos os números desde 1 até esse numero, excluindo todos os números múltiplos de 2 e 3.

Exemplo de execução: Qual o numero? 11 1 5 7 11

b) Pirâmide de Números (ciclos). Escreva um programa que dado um valor N introduzido pelo utilizador coloque os seguintes números no ecrã. Exemplo de execução:

N=3
1
2 1
3 2 1

N=6
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1

N=10
1
2 1
3 2 1
4 3 2 1
5 4 3 2 1
6 5 4 3 2 1
7 6 5 4 3 2 1
8 7 6 5 4 3 2 1
9 8 7 6 5 4 3 2 1
10 9 8 7 6 5 4 3 2 1