UNIVERSIDADE DA BEIRA INTERIOR

Programação – LEI + LMA **1º Semestre**

Exame Época Recurso (16 val) **2h + 15 min** **06/02/2023**

1. [1.5 val]

Escreva uma **instrução de atribuiçao** em linguagem C para cada uma das seguintes acções:

(a) atribua à variável **B** o dobro a parte inteira do valor da variável do tipo real **X**

(b) atribua à variável **P** o valor **5** se o valor da variável do tipo inteiro **N** for par e **-5** se **N** for ímpar

(c) atribuir à variável **SOMA** a soma dos algarismos das dezenas e das unidades do valor da variável do tipo inteiro **A** (Ex: A = 3572 ==> SOMA = 9 (7+2))

Supondo que **X = -10**, **Y = 10** e **Z = 5**, **indique** a **ordem de cálculo dos operadores** e **determine** o **valor** de **cada** uma das seguintes **expressões**. Justifique, apresentando os cálculos efetuados.

	1		**2**		**3**		**4**		**5**		**6**	
(d)	Y	**<**	Z	*****	(3	**+**	X)	**&&**	X	**>=**	-2	*****	Z
(e)	(Z	**+**	X)	*****	10.0	*****	(11	**/**	3	**/**	3)		
(f)	Z	**+**	(2	**+**	Y)	*****	4	**/**	(Z	**-**	X)		

Sugestão de resposta para cada alínea de (d) a (f) (exemplo):

- ordem de cálculo: 3, 5, 2, 1, 4
- valor: 34 (e apresentar cáculos)

2. [2.5 val]

Construa um algoritmo, usando um fluxograma, que:

- peça ao utilizador que **insira/leia** vários números **inteiros** (**termina** quando inserir o **zero** (0));
- determine a **quantidade** de números **positivos** inseridos e a **soma** dos números **positivos** inseridos;
- mostre os **resultados** obtidos antes (quantidade e soma).

NOTA: Não usar arrays

3. [2.0 val]

Construa um **programa em C** que:

- peça ao utilizador que **insira/leia** vários números **inteiros** (**termina** quando inserir o **zero** (0));
- determine a **quantidade** de números **positivos** inseridos e a **soma** dos números **positivos** inseridos;
- mostre os **resultados** obtidos antes (quantidade e soma).

NOTA: Não usar arrays

4. [2.5 val]

Implementar um **subprograma** em C que dado dois números inteiros positivos não nulo **N1** e **N2** (N1, N2 > 0), com **N1 < N2** (N1 e N2 parâmetros do subprograma), **determine e devolva** como resultados a **soma dos números pares** e a **soma dos números ímpares** existentes entre **N1** e **N2**, inclusivé (em {N1, …, N2}).

5. [1.0 val]

Considere as seguintes declarações de variáveis:

int *V, **W, *X;

e que **sizeof(int) = 4** e **sizeof(int *) = 8**.

(esquema de um bloco de memória)

	...	
100500	110172	**X**
	...	
100700	110168	**V**
	...	
110160	100	
110164	900	
110168	500	
110172	1000	
110176	1500	
110180	50	
	...	
110500	100700	**w**
	...	

Usando os valores contidos no esquema de um bloco de memória dado ao lado, indique, justificando, os valores de cada uma das seguintes expressões:

a) *(X + 2)

b) V - 4

c) W[0]

d) &(*(X - 1))

e) *X - 1

f) W + 2

g) *(*W - 2)

h) *V + 3

NOTA: Caso o valor da expressão não exista ou seja desconhecido, deve responder "**Indefinido**".

6. [2.5 val]

Implementar um **subprograma** em C que dado um array de 1 dimensão **X** com **N** números inteiros (X e N são parâmetros do subprograma), **determine e devolva** o **segundo menor número** que existe em **X**. Exemplo: X = [21 7 2 21 -14 2 -14 23], segundo menor número é o **2** (o menor número é -14)

7. [4.0 val]

Considere a seguinte definição de um tipo estrutura associada aos dados de uma pessoa:

```
typedef struct {
    int     numCC;      // Número de Cartão de Cidadão
    int     anoNasc;    // Ano de nascimento (ex: 1986)
    float   altura;     // Altura em metros (exemplo: 1.85)
} PESSOA;
```

Considere também o seguinte subprograma que se encontra já implementado na biblioteca "Exame.h":

void maiorAltura (PESSOA X[], float *maior);

que **devolve** como resultado o **maior valor do campo altura** dos elementos (registos) do array **X**.

Seja o ficheiro de texto **"Pessoas.txt"** com dados até no máximo 500 pessoas; cada linha do ficheiro guarda os dados de uma única pessoa (por esta ordem): número de CC, ano de nascimento e altura.

Implemente um **programa em C** que realize as seguintes acções (pela ordem indicada):

- com os dados do ficheiro **"Pessoas.txt"**, **construa** um **array de 1 dimensão X** do tipo **PESSOA** com **todos** os dados do ficheiro,
- determine o **maior valor** da **altura** dos elementos do array **X**, **maior** (usar subprograma dado),
- peça ao utilizador um número real **ALT** menor ou igual a **maior** (**0 < ALT ≤ maior**),
- **guarde** no ficheiro **"Altura.txt"** todos os elementos do array **X** cuja valor do campo **altura** é maior ou igual a **ALT** (≥ ALT)