Estruturas, Ponteiros, Memória Dinâmica e Recursividade

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

```
typedef struct {
   int  numAluno;        // número de aluno ({ 70000, …, 75000 })
   float  notasMTP[2];   // notas dos mini-testes práticos ([0.0, 2.0])
   float  notasTE[2];    // notas dos testes escritos ([0.0, 8.0])
   int  notaFinal;       // nota final ({ 0, …, 20 })
} ALUNO;
```

Crie uma pasta no seu computador e descarregue da página web da disciplina os ficheiros **"Aleatorio.h"**, **"Introducao.h"** e **"IntroducaoMain.c"** para esta pasta. Na resolução dos exercícios que se seguem, deve implementar todas as funções pedidas na biblioteca "**Introducao.h**" e o programa principal (main) no ficheiro "**IntroducaoMain.c**".

1. Implementar uma **função** que crie um array 1D do tipo ALUNO (usar gestão dinâmica da memória) com dados gerados aleatoriamente (funções da biblioteca "Aleatorio.h"). O tamanho do array deve ser um valor entre 1 e 30 e gerado aleatoriamente. Gerar números aleatórios para preencher os seguintes campos do array: **numAluno** (não pode haver valores iguais), **notasMTP** e **notasTE** (ver limites para cada um dos campos). Usar o seguinte protótipo:

   ALUNO *criarArray (int *N);

2. Implementar uma **função** que dado um array do tipo ALUNO com N elementos, mostre no monitor todos os elementos do array (cada elemento/registo numa linha: numAluno – notasMTP – notasTE - notaFinal). Usar o seguinte protótipo:

   void mostrarArray (ALUNO *A, int N);

3. Implementar um **programa** que, usando as funções implementados em 1 e 2, realize as seguintes ações pela ordem indicada:
   a) construir um array 1D do tipo ALUNO com dados gerados aleatoriamente,
   b) mostre no monitor os dados do array construído antes.

4. Implementar uma **função** que dado um array do tipo ALUNO com N elementos, atualize o campo "**notaFinal**" do array (soma dos valores dos campos "**notasMTP**" e "**notasTE**"). Usar o seguinte protótipo:

   void atualizarArray (ALUNO *A, int N);

   Teste acrescentando ao programa implementado em 3 e depois mostre novamente o array.

5. Implementar uma **função** que dado um array do tipo ALUNO com N elementos, **guarde** os valores dos campos "**numAluno**" e "**notaFinal**" de cada elemento do array no ficheiro "**NotasFinais.txt**" (os 2 valores numa linha: numAluno - notaFinal). Usar o seguinte protótipo:

   void ficheiroNotasFinais (ALUNO *A, int N);

   Teste acrescentando ao programa implementado em 3.

6. Implementar uma **função iterativa** que dado um array do tipo ALUNO com N elementos, **devolva** o maior valor do campo "**notaFinal**" do array. Usar o seguinte protótipo:

int maiorNotaFinal_IT (ALUNO *A, int N);

Teste acrescentando ao programa implementado em 3.

7. Implementar uma **função recursiva** que dado um array do tipo ALUNO com N elementos, **devolva** o maior valor do campo "**notaFinal**" do array. Usar o seguinte protótipo:

int maiorNotaFinal_REC (ALUNO *A, int N);

Teste acrescentando ao programa implementado em 3.

8. Implementar uma **função** que dado um array do tipo ALUNO A com N elementos, devolva um outro array do tipo ALUNO com os elementos de A que obtiveram aprovação (notas finais maiores ou iguais a 10). Usar o seguinte protótipo:

void arrayAprovados (ALUNO *A, int N, ALUNO **AP, int *numAP);

Teste acrescentando ao programa implementado em 3.

9. Implementar uma **função recursiva** que dado um array do tipo ALUNO com N elementos e um número inteiro K, devolva a quantidade de alunos (registos) cuja valor do campo "**notaFinal**" é maior ou igual ao valor de K. Usar o seguinte protótipo:

int quantNotaFinalMaiorIgualK (ALUNO *A, int N, int K);

Teste acrescentando ao programa implementado em 3, para realizar as seguintes ações:

a) determinar e mostrar a quantidade de elementos do array com valor no campo "notaFinal" igual ao maior valor daquele campo,

b) determinar e mostrar a quantidade de elementos do array com valor no campo "notaFinal" maior ou igual a 10.

10. Implementar uma **função** que dado um array do tipo ALUNO com N elementos, **devolva** um array de inteiros com os valores do campo "**notaFinal**" de todos os elementos do array. Usar o seguinte protótipo:

int arrayNotasFinais (ALUNO *A, int N, int **V);

Teste acrescentando ao programa implementado em 3.

11. Implementar uma **função** que dado um array 1D com N números inteiros, **devolva** a **quantidade** e a **soma** dos elementos do array que são maiores ou iguais ao valor de K. Usar o seguinte protótipo:

void somaElementosArrayInt (int *V, int N, int K, int *soma, int *quant);

Teste acrescentando ao programa implementado em 3, para determinar

a) a média dos valores do campo "**notaFinal**" dos elementos do array do tipo ALUNO,

b) a média dos valores do campo "**notaFinal**" maiores ou iguais a 10 (≥ 10) dos elementos do array do tipo ALUNO, e

c) a média dos valores do campo "**notaFinal**" menores que 10 (< 10) dos elementos do array do tipo ALUNO