Ficha de Trabalho N.º 6

Objectivos: Estruturas dinâmicas: *Hashing*

Os exercícios propostos visam criar um programa que faça a gestão de um conjunto de pessoas por faixas etárias, com intervalos de 10 anos, entre os 0 e 100 anos. Esta gestão deve ser feita usando um vector de *hashing* para facilitar a manipulação das diferentes listas de pessoas.

1. Defina uma estrutura **Pessoa**, associando-lhe o tipo de dados ***ptPESSOA**, com os dados:

a) Nome (20 caracteres);
b) Idade em anos (inteiro);
c) Peso em quilos (número real);
d) Altura em metros (número real);

2. Defina uma estrutura **Elemento**, associando-lhe o tipo de dados ***ptELEM**, adequada para a construção de uma lista ligada de elementos e com um campo do tipo **ptPESSOA**.

3. Defina uma estrutura **Lista**, associando-lhe o tipo de dados ***ptLISTA**, com os dados:

a) Número de elementos (inteiro);
b) Elemento inicial da lista.

4. Defina uma estrutura **Grupo**, com os dados:

a) Faixa etária (inteiro);
b) Lista de pessoas (**ptLISTA**);

5. Defina uma estrutura **Hashing**, associando-lhe o tipo de dados ***ptHASHING**, com um campo vector que contém 10 elementos do tipo **GRUPO**, que irão representar faixas etárias distintas dos 0 aos 100 anos em intervalos consecutivos de 10 anos.

6. Escreva uma função **ptLISTA criarLista()** que crie (aloque e inicialize) uma lista de elementos.

7. Escreva uma função **ptELEM criar_elemento()** que crie (aloque e inicialize) um novo elemento.

8. Escreva uma função **void ler_elemento(ptELEM ele_novo)** que registe os dados de um dado elemento

9. Escreva uma função **int comparar_elementos(ptELEM A, ptELEM B)** que compare dois elementos, recorrendo ao campo idade da pessoa associada.

10. Escreva uma função **int elementos_iguais(ptELEM A, ptELEM B)** que verifique a igualdade entre dois elementos.

11. Escreva uma função **void inserir_elemento_ordenado(ptLISTA L, ptELEM ele_novo)** que introduza um elemento na lista, garantido a sua ordenação através do campo idade da pessoa.

12. Escreva uma função **ptELEM pesquisar_elemento(ptLISTA L, ptELEM ele_pesquisa)** que verifique se um determinado elemento pertence a uma lista utilizando uma abordagem iterativa.

13. Escreva uma função **void libertar_elemento(ptELEM ele_libertar)** que liberte o espaço alocado para um dado elemento e respectivas informações associadas.

14. Escreva uma função **ptELEM remover_elemento(ptLISTA L, ptELEM ele_remover)** que remova um determinado elemento de uma lista.

15. Escreva uma função **void mostrar_elemento(ptELEM ele_mostrar)** que mostre os dados de uma pessoa.

16. Escreva uma função **void mostrar_ordenado(ptLISTA L)** que mostre os dados de todos os elementos de uma lista pela ordem actual.

17. Escreva uma função **ptHASHING criar_vector_hashing()** que crie (aloque) um vector de *hashing*.

18. Escreva uma função **void inicializar_vector_hashing(ptHASHING H)** que inicialize um vector de *hashing*.

19. Escreva uma função **int posicao_hashing_elemento(ptELEM E)** que calcule a posição de *hashing* correcta para um dado elemento.

20. Escreva uma função **int validar_posicao_hashing(int pos)** que valide uma posição de *hashing* calculada através da função do ponto anterior.

21. Escreva uma função **void inserir_elemento_hashing(ptHASHING H, ptELEM E)** que introduza um dado elemento num vector de *hashing*.

22. Escreva uma função **ptELEM remover_elemento_hashing(ptHASHING H, ptELEM ele_remover)** que remova um elemento de um vector de *hashing.*

23. Escreva uma função **ptELEM pesquisar_elemento_hashing(ptHASHING H, ptELEM ele_pesquisa)** que pesquise por um elemento num vector de *hashing*.

24. Escreva uma função **void mostrar_elementos_hashing(ptHASHING H)** que imprima os dados de todos os elementos que constam do vector de *hashing*.

25. Escreva uma função **int menu_principal()** que permita ao utilizador escolher entre as diversas funcionalidades do programa.

```
(1) Inserir um novo elemento no vector de hashing
(2) Retirar um elemento do vector de hashing
(3) Mostrar os elementos do vector de hashing
(4) Pesquisar um elemento no vector de hashing
(0) SAIR
```

26. Escreva a função principal **void main()** de modo a integrar as funções elaboradas anteriormente.