

EXERCÍCIOS DE FIXAÇÃO Nº 07
Alocação Dinâmica de Memória

- 1) Escreva uma função que recebe um vetor *float* **v** e sua capacidade **n**, e retorne o endereço de um vetor alocado dinamicamente, cujo conteúdo seja o mesmo de **v**, ou seja, a função retorna um *clone* do vetor **v**. Faça o programa principal com a entrada de dados (ou um vetor fixo), chame a função e mostre o vetor resultante na tela. Protótipo da função:

`float *clone(float *v, int n);`

- 2) Escreva uma função que recebe como parâmetros uma *string* **s** e um inteiro **n**, e retorna nova *string* nova contendo **s** repetida **n** vezes. Por exemplo, **s** = "Abc" e **n** = 4 tem como resultado a *string* "AbcAbcAbcAbc". Faça o programa principal chamando a função. Protótipo da função:

`char *repetidor(char *s, int n);`

- 3) Escreva um programa que aloca dinamicamente um vetor do tipo *float* e realiza a entrada de dados. Em seguida, o programa deve calcular a *média* dos valores do vetor e alocar dinamicamente um novo vetor contendo somente os valores maiores ou iguais à média. O processo pode ser feito usando *malloc()*, ou seja, fazendo a contagem, alocação e cópia dos valores. Outra alternativa consiste em usar *realloc()* para ir aumentando o espaço alocado à medida que os valores vão sendo encontrados.

- 4) Faça um programa que leia uma certa quantidade de inteiros que são armazenados num vetor **v**. A quantidade deve ser definida pelo usuário, e o programa aloca espaço para **v**. O programa deve armazenar os valores positivos em um vetor **vp** e o valores negativos no vetor **vn**. Como as quantidades de valores positivos e negativos são desconhecidas, o espaço para **vp** e **vn** deve ser alocado dinamicamente. Os vetores **vp** e **vn** não devem conter zeros. Ao final, imprima os três vetores. Pode ser feito com *malloc()* ou com *realloc()*.

- 5) Escreva uma função que realiza a **união** entre dois conjuntos de inteiros contidos nos vetores **v1** e **v2**. A função recebe os vetores e suas respectivas capacidades (**n1** e **n2**) como parâmetros de entrada e retorna o endereço do vetor alocado (contendo a união entre **v1** e **v2**). Além disso, há um parâmetro passado por referência (ponteiro **p3**), que serve para "retornar" a capacidade do vetor gerado. Faça o programa principal invocando a função (a estrutura do programa é semelhante ao exemplo dado em aula – *intersecção*). Protótipo da função:

`int *uniao(int *v1, int n1, int *v2, int n2, int *p3);`