Décimo Laboratório de ECOP11

UNIFEI - Universidade Federal de Itajubá
Prof. Gabriel Cirac Mendes Souza (gabrielcirac@unifei.edu.br)

Assunto: Recursão

Vamos codificar o algoritmo de Busca Binária em C. Esse tipo de busca funciona apenas para vetores ordenados. Nela, testamos inicialmente se o valor procurado está exatamente no meio do vetor. Como o vetor se encontra ordenado, se o valor no meio do vetor for maior do que o procurado, ele se encontra na primeira metade. Caso contrário, se encontra na segunda metade. Aí, aplicamos a busca novamente, utilizando somente a parte do vetor que nos interessa.

1) (10 pontos) Declare um vetor com 25 posições, na função main.

2) (20 pontos) Escreva uma função `inicializa_vetor`, que será responsável pela geração dos 25 números aleatórios que compõem o vetor (os números gerados devem estar entre 0 e 100). Ela recebe um vetor de inteiros como parâmetro, seu tamanho, e não retorna nada (é do tipo `void`).

3) (10 pontos) Escreva uma função `imprime_vetor`, que será responsável pela impressão dos valores do vetor gerado. Ela recebe um vetor de inteiros como parâmetro, seu tamanho, e também não retorna nada (é do tipo `void`).

4) (10 pontos) Copie o código de ordenação disponibilizado em conjunto com o guia (`bubble.c`), e realize a chamada da função `bubble`. Para que a busca binária seja realizada, primeiro é necessário que o vetor seja ordenado.

5) (40 pontos) Escreva a função recursiva `busca_binaria`.

    a. Ela irá retornar um número inteiro, referente à primeira posição onde o número buscado for encontrado, ou -1 caso não seja encontrado.
    b. Seus parâmetros serão o próprio vetor, um inteiro para o elemento buscado, a posição do vetor onde iniciaremos nossa busca, e a posição final, onde terminaremos a busca.

```c
int busca_binaria(int vet[], int chave, int ini, int fin)
```

    c. O algoritmo consiste em:
        i. Encontrar a posição central do vetor (`int meio`).
        ii. Verificar se o elemento na posição central é o buscado. Se sim, retorne `meio`.
        iii. Caso contrário:
            - Se `(fin == ini)`, retorne -1, pois o elemento não foi encontrado.
            - Senão, verifique se o elemento na posição `meio` é maior ou menor do que a chave.
                - Se for maior, faça uma **chamada recursiva** da função `busca_binaria` especificando os limites `meio+1` e `fin`: Busca na metade superior.

- Se for menor, faça uma chamada recursiva da função `busca_binaria` especificando os limites `ini` e `meio-1`: Busca na metade inferior.

6) (10 pontos) Copie a função main() abaixo e verifique o funcionamento do programa:

```c
int main()
{
    int vetor[25], chave, posicao;

    // Gera valores aleatorios e imprime vetor
    srand(time(NULL));
    inicializa_vetor(vetor, 25);
    printf("Vetor Gerado: \n");
    imprime_vetor(vetor, 25);

    // Ordena vetor e imprime
    bubble(vetor, 25);
    printf("\nVetor Ordenado: \n");
    imprime_vetor(vetor, 25);

    // Pede repetidamente numeros para serem buscados
    do
    {
        printf("\nEntre com um elemento para busca ou -1 para sair:");
        scanf("%d", &chave);

        if(chave > -1)
        {
            posicao = busca_binaria(vetor, chave, 0, 24);

            if(posicao == -1)
                printf("Elemento nao encontrado. \n");
            else
                printf("Elemento encontrado na posicao %d. \n", posicao);
        }

    } while(chave != -1);


    return 0;
}
```