Universidade Federal do ABC
Algoritmos e Estruturas de Dados II 3^o Quadrimestre 2018

Prof. Carlos da Silva dos Santos

Aula prática 01 – Busca Linear e Binária.

1 - Escreva uma função usando a linguagem `C` que realiza a *busca sequencial* de um valor em um vetor de inteiros. Você pode usar a declaração abaixo para a função:

```
int seq_search(int array[], int array_size, int key);
```

Na declaração acima, `array` é um vetor contendo uma sequência de números inteiros; `array_size` é o número de elementos do vetor e `key` é o valor que estamos buscando em `array`. A função deve devolver o índice do elemento, caso esse seja encontrado. Caso contrário, a função deve devolver o valor -1.

2 - Escreva uma função usando a linguagem `C` que realiza a *busca binária* de um valor em um vetor *ordenado* de inteiros. Você pode usar a declaração abaixo para a função:

```
int binary_search(int array[], int array_size, int key);
```

3 - Escreva um programa para testar as duas funções implementadas anteriormente. Use diferentes tamanhos de entrada e teste o caso em que a o valor buscado está presente no vetor e o caso em que o valor não está presente. Tente criar instâncias que sejam "difíceis" para os algoritmos (isto é, que acionem o comportamento de pior caso).

4 - Inclua no seu programa a bibliteca `time.h` que permite medir tempos de execução de trechos de código. O exemplo abaixo mostra como medir o tempo de execução de uma função, guardando na variável `delta`.

```
clock_t t_inicio, t_fim;
...
t_inicio = clock();
result = seq_search(data, 10, key1);
t_fim = clock();
delta = 1e6*(double)(t_fim - t_inicio)/CLOCKS_PER_SEC;
// delta medido em ns (nano segundos)
```

Meça o tempo de execução de cada função, para diferentes tamanhos de entrada.

5 - A busca binária somente funciona corretamente caso a entrada esteja ordenada. Alguém sugere modificar a função para que ocorra uma verificação da entrada, garantindo que o vetor esteja ordenado antes da realização da busca. Como isso afetaria a complexidade da função?