Introdução à Ciência da Computação II SSC0503

Professor: Adenilso da Silva Simão
PAE: Jorge Francisco Cutigi
Monitor: Luis Felipe Jorge

Atividade avaliativa 05

Métodos de ordenação: Shell e Quick

Data de entrega: 18/10/2020

Instruções:

Os trabalhos devem ser entregues no run.codes, na disciplina com código SHEL. O exercício será feito em grupo formado por 3 ou 4 alunos, o qual será definido por sorteio. O grupo sorteado será indicado no e-disciplinas. Todos os integrantes devem submeter o mesmo código. O grupo deve se "reunir" ao menos uma vez pelo google meet para discutir/resolver o problema e deve gravar a sessão. Além do código, deve ser postado o link para a vídeo da sessão gravada (no máximo 5 minutos) no fórum específico no e-disciplinas. O vídeo deve estar aberto para visualização de todos os alunos (compartilhar como "Qualquer pessoa da USP pode visualizar"). Caso tenham problemas com o grupo (por exemplo, não consigam contato), por favor, avisar até 14/out.

Atividade:

Faça um programa em C que leia o número *N* de elementos de um vetor *V*. Após isso, leia os *N* elementos de *V*. O programa deve ordenar todos os subvetores de *V* utilizando os métodos Shell Sort e Quick Sort. Para cada subvetor de *V*, deve-se imprimir qual método fez menos operações (comparação + cópias), imprimindo Q quando for o Quick Sort, S quando for o Shell Sort e o caractere hífen quando o número de contagens for igual. O Shell deve utilizar os gaps no formato 2^k - 1, e o Quick Sort deve utilizar o elemento do meio como pivô (como os códigos vistos em aula).
Por exemplo, considere o seguinte vetor *V*: [3, 6, 5, 2]. Os subvetores de V são: *V1*: [3], V2: [3, 6], V3: [3, 6, 5], e V4 = [3, 6, 5, 2]. Para V1, o número de operações é o mesmo para os dois métodos. Para V2, V3 e V4, o Shell Sort realiza menos operações. Portanto, seu programa deve imprimir: `- S S S`

Utilize o código abaixo para o Shell sort:

```
void shell(int v[], int n) {
    int gap = 1;
    while(gap <= n) {
        gap *= 2;
```

```
    }
    gap = gap / 2 - 1;
    while(gap > 0) {
        int i;
        int j;
        for (int k = 0; k < gap; k++){
            for (int i = k + gap; i < n; i += gap)
{
                int x = v[i];
                j = i - gap;
                while(j >= 0 && v[j] > x) {
                    v[j+gap] = v[j];
                    j-=gap;
                }
                v[j+gap] = x;
            }
        }
        gap /= 2;
    }
}
```

Observações importantes:

1. Somente comparações e trocas que envolvam elementos do vetor devem ser contadas. Por exemplo, comparações do tipo **`i < fim`** não envolvem elementos do vetor, ao contrário de comparações do tipo **`x > vetor[i]`**, que envolve o vetor a ser ordenado.
2. Considerar o conceito de "curto circuito" em C. Por exemplo, na expressão **`if (i > 0 && v[i] > x)`**, quando **`i > 0`** é falso, a comparação **`v[i] > x`** não é realizada.

Exemplo de entrada e saída:

Entrada	Saída
4 3 6 5 2	- S S S
12 1 2 3 4 5 6 7 8 9 0 9 8	- S S S S S S Q Q S S S