2ª Avaliação de Programação de *Scripts* – 2ª Parte

Redes de Computadores – Campus da UFC em Quixadá – 2019.2

Instruções: dentro da pasta *prova* no servidor, crie a pasta *prova02* e coloque os *scripts* das questões.

3ª Questão) O nome do *script* deve ser questao03.sh. Esta questão é uma continuação da questão 02 da 1ª Parte. Além da função *Bubble Sort* você irá agora implementar uma função para o método de ordenação *Quick Sort* (https://pt.wikipedia.org/wiki/Quicksort). Agora, depois de ordenar o vetor, vamos fazer uma análise de desempenho. Você irá gerar um vetor aleatório e medir o tempo de execução de cada uma das funções de ordenação.

```
#!/bin/bash
# Criar e imprimir o vetor aleatório.
...
# Capturar o tempo inicial do Bubble Sort
inicio_bs=`date +%s`
# Ordenar por Bubble Sort
...
# Capturar o tempo final do Bubble Sort
fim_bs=` date +%s`
duracao_bs=`expr $fim_bs - $inicio_bs`
# Imprime o vetor ordenado
...
echo “Duração da ordenação Bubble Sort: $duracao_bs”

# Capturar o tempo inicial do Quick Sort
inicio_qs=`date +%s`
# Ordenar por Quick Sort
...
# Capturar o tempo final do Quick Sort
fim_qs=` date +%s`
duracao_qs=`expr $fim_qs - $inicio_qs`
# Imprime o vetor ordenado
...
echo “Duração da ordenação Quick Sort: $duracao_qs”
```

O objetivo é entender qual é o algoritmo mais rápido. Faça um teste com N=10, N=100 e N=1000. Não é preciso reportar os valores na prova, mas o tempo de execução do *Quick Sort* deve ser menor do que o *Bubble Sort*.

4ª Questão) O nome do *script* deve ser questao04.sh. Você deve criar uma função chamada *multTable*. Essa função deve receber um parâmetro e imprimir a tabela de multiplicação até esse número. A saída deve ter o seguinte formato:

$ ./questao04.sh 5

	1	**2**	**3**	**4**	**5**
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20
5	5	10	15	20	25

Os índices estão em vermelho e em negrito. Os elementos da diagonal estão sublinhados.

5ª Questão) O nome do *script* deve ser questao05.sh. Esse *script* deve receber dois parâmetros.

$ ./questao05.sh [DIR1] [DIR2]

O primeiro parâmetro (vamos chamar de **DIR1**) é o caminho para um diretório que contém apenas arquivos, sem subdiretórios. Para cada arquivo em DIR1 o *script* deve recuperar a data de modificação, incluindo o ano, mês e dia. O arquivo então deve ser copiado para **DIR2** (informado no segundo parâmetro), mas deve ser colocado em um diretório no formato DIR2/YYYY/MM/DD, no qual YYYY, MM e DD são o ano, o mês e o dia da data de modificação. Por exemplo, considere que o arquivo DIR1/teste.txt tem a data de modificação "2018-10-16 13:28:52". Ele deve ser copiado para o diretório DIR2/2018/10/16. Arquivos com a mesma data de modificação devem ficar no mesmo diretório.

Para testar seu script, você pode usar o programa abaixo. Ele cria aleatoriamente um diretório com arquivos com datas de modificações diferentes.

```
#!/bin/bash
mkdir teste;
for i in $(seq 10)
do
  ano=$(shuf -e 200{1..9} 201{0..8} | tail -n 1);
  mes=$(shuf -e 0{1..9} 10 11 12 | tail -n 1);
  dia=$(shuf -e 0{1..9} 1{0..9} 2{0..9} 30 | tail -n 1);
  touch -m -t ${ano}${mes}${dia}0000 teste/arquivo$i.txt
done
```