

## Exercicio Redis

334409 – Charles Guimarães Cavalcante

### Trabalho 1

**Construa uma estrutura de dados que capture os seguintes nomes:**

Bruce, Robert, Fred, Jim, Chimbinha, Ronnie , James, Chris

Características da Lista: Armazenar somente nomes distintos.

Criei uma chave chamada “nomes” do tipo SET para armazenar os dados, pois é uma lista que tem como característica armazenar valores distintos.

Comandos:

```
SADD nomes Bruno
```

```
SADD nomes Robert
```

```
SADD nomes Fred
```

```
SADD nomes Jim
```

```
SADD nomes Chimbinha
```

```
SADD nomes Ronnie
```

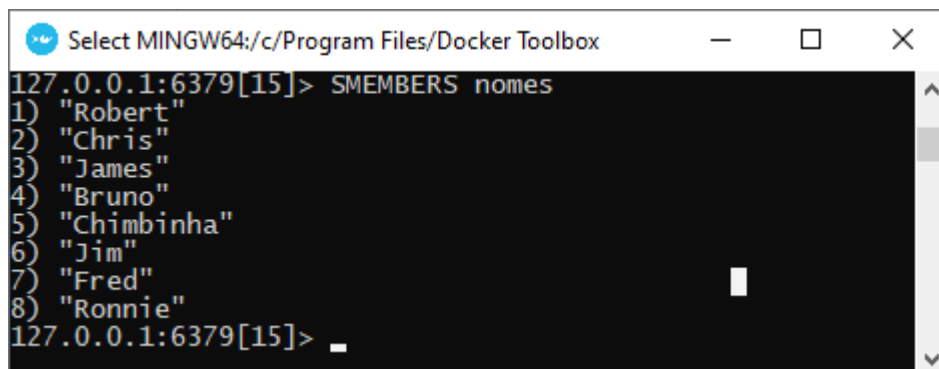
```
SADD nomes James
```

```
SADD nomes Chris
```

**Apresente uma visão da lista armazenada, depois de feitas 8 entradas.**

Retorno dos dados com o comando:

```
SMEMBERS nomes
```



```
Select MINGW64:/c:/Program Files/Docker Toolbox
127.0.0.1:6379[15]> SMEMBERS nomes
1) "Robert"
2) "Chris"
3) "James"
4) "Bruno"
5) "Chimbinha"
6) "Jim"
7) "Fred"
8) "Ronnie"
127.0.0.1:6379[15]> _
```

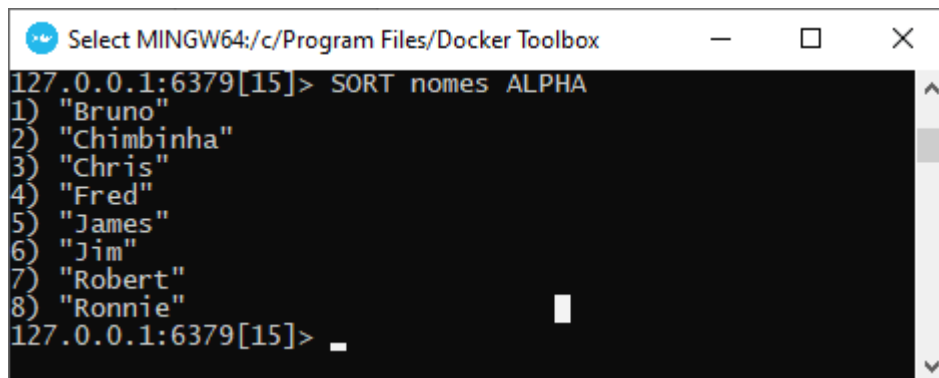
**Qual a ordem de saída dessa lista? Está respeitada alguma ordem?**

A saída desta lista apresenta por padrão os dados na ordem de inserção.

**Essa característica deve ser vista como um problema? Justifique.**

Esta característica não é um problema, pois caso precisemos dos dados ordenados podemos utilizar o comando abaixo:

`SORT nomes ALPHA`

A terminal window titled "Select MINGW64:/c/Program Files/Docker Toolbox" shows the command `127.0.0.1:6379[15]> SORT nomes ALPHA` being executed. The output is a list of names in alphabetical order: `1) "Bruno"`, `2) "Chimbinha"`, `3) "Chris"`, `4) "Fred"`, `5) "James"`, `6) "Jim"`, `7) "Robert"`, and `8) "Ronnie"`. The prompt `127.0.0.1:6379[15]>` is visible at the bottom.

## Trabalho 2

Pensando em nosso case contínuo da plataforma virtual MarketPlace de vendas.

**Construa duas estruturas de Dados:**

- Para atender dados de Sessão:

Utilizei a estrutura de HASH para armazenar os dados da sessão:

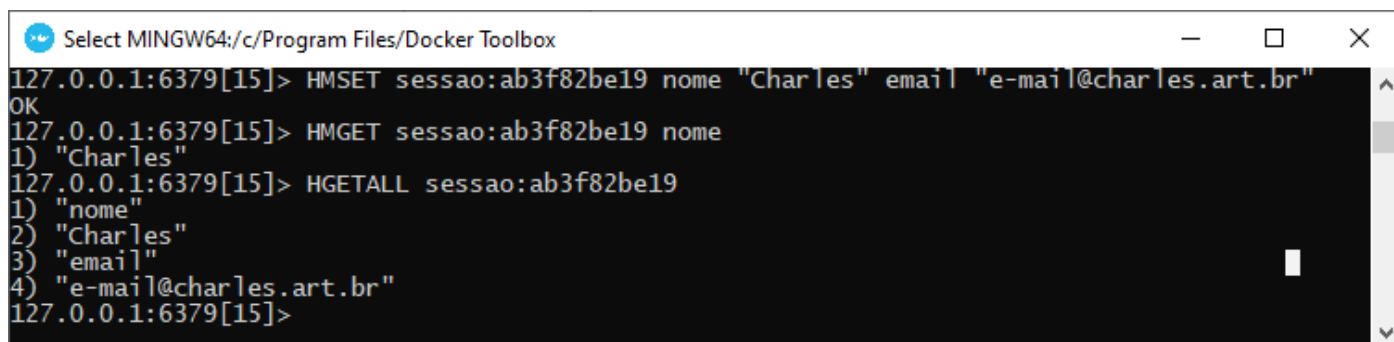
```
HMSET sessao:ab3f82be19 nome "Charles" email "e-mail@charles.art.br"
```

Para retorna os da sessão:

```
HMGET sessao:ab3f82be19 nome
```

Para retorna todos os dados da sessão:

```
HGETALL sessao:ab3f82be19
```

A terminal window titled "Select MINGW64:/c/Program Files/Docker Toolbox" shows three commands being executed. First, `HMSET sessao:ab3f82be19 nome "Charles" email "e-mail@charles.art.br"` returns `OK`. Then, `HMGET sessao:ab3f82be19 nome` returns `1) "Charles"`. Finally, `HGETALL sessao:ab3f82be19` returns a list of four items: `1) "nome"`, `2) "Charles"`, `3) "email"`, and `4) "e-mail@charles.art.br"`. The prompt `127.0.0.1:6379[15]>` is visible at the bottom.

- Para atender dados de Carrinho de compras:

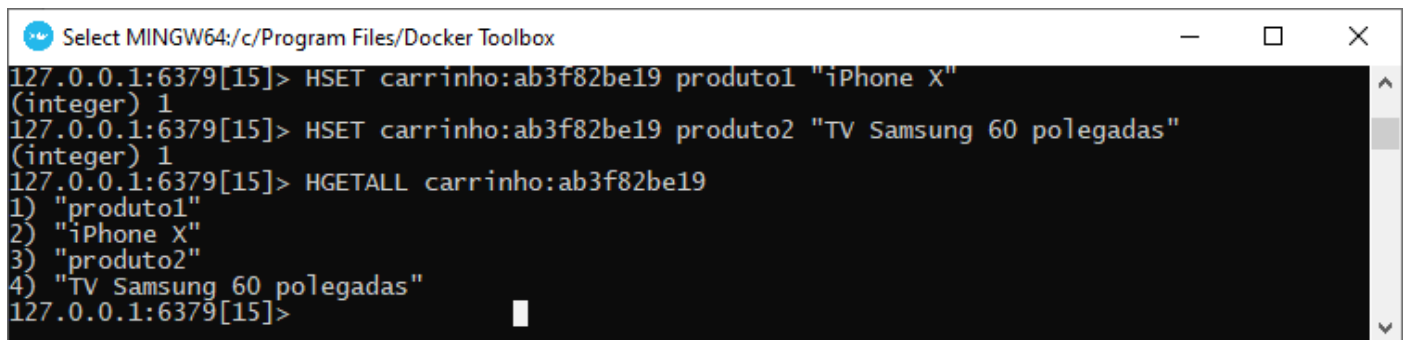
Utilizei também a estrutura de HASH para armazenar os dados da sessão, porém utilizando o comando HSET para adicionar um item de cada vez:

```
HSET carrinho:ab3f82be19 produto1 "iPhone X"
```

```
HSET carrinho:ab3f82be19 produto2 "TV Samsung 60 polegadas"
```

Para retorna todos os dados do carrinho:

```
HGETALL carrinho:ab3f82be19
```



```
Select MINGW64:/c:/Program Files/Docker Toolbox
127.0.0.1:6379[15]> HSET carrinho:ab3f82be19 produto1 "iPhone X"
(integer) 1
127.0.0.1:6379[15]> HSET carrinho:ab3f82be19 produto2 "TV Samsung 60 polegadas"
(integer) 1
127.0.0.1:6379[15]> HGETALL carrinho:ab3f82be19
1) "produto1"
2) "iPhone X"
3) "produto2"
4) "TV Samsung 60 polegadas"
127.0.0.1:6379[15]>
```