

## Arquitetura de Computadores 2022/23

### Ficha 2

**Tópicos:** Programas em C para manipulação de ficheiros. 1º Trabalho Prático

### Programas em C para manipulação de ficheiros

---

- Criar, compilar e executar um programa em C (*hello.c*) que escreva no *standard output* a mensagem “Hello world”. Pode usar um editor de texto do ambiente gráfico como o medit ou o gedit (Text editor), ou então um no terminal como o nano.

- Crie um programa para copiar ficheiros. O programa deve receber 2 nomes de ficheiro como argumentos, como em

```
./copial nome_ficheiro_original nome_ficheiro_copiado
```

O programa deve utilizar as chamadas ao sistema `open`, `read`, `write` e `close` e deve copiar 1024 bytes de cada vez.

- Faça uma nova versão do programa para copiar ficheiros de texto organizados em linhas.. O programa deve receber 2 nomes de ficheiro como argumentos, como em

```
./copia2 nome_ficheiro_original nome_ficheiro_copiado
```

O programa deve utilizar as funções da biblioteca `stdio` do C `fopen`, `fclose`, `fgets` e `fputs`,

- Crie um programa em C que compara o conteúdo de 2 ficheiros linha a linha, tal como no exemplo

```
./compara nome_ficheiro_1 nome_ficheiro_2
```

Para cada linha diferente deve ser mostrado o número da linha e o conteúdo das duas linhas diferentes. Exemplo de um output em que há dois ficheiros diferentes:

```
#1: 66b8a3879d7c13017ce07626f012684895c25670 teste/a/1
    f690f7e924614995dd26bf38cda79a25d308eef0 teste/a/1
```

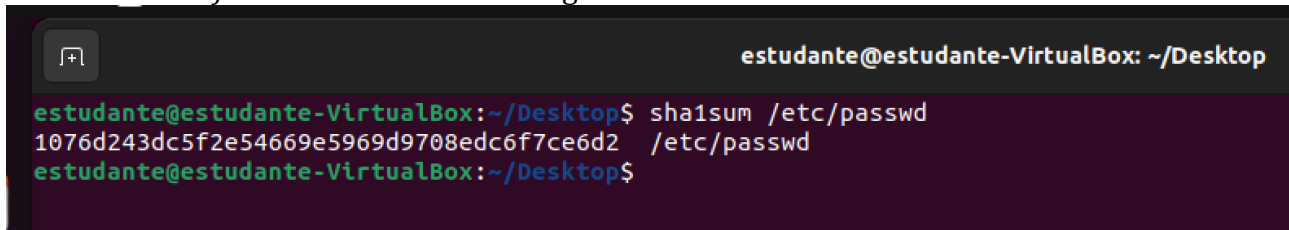
```
#5: d066b8ef276f78052c9ffecd3413bce8e6562705 teste/b/3
    f690f7e924614995dd26bf38cda79a25d308eef0 teste/b/3
```

### 1º Trabalho Prático

---

Uma operação importante para garantir que o software que existe num computador não foi alterado maliciosamente é comparar cada ficheiro com aquele que foi originalmente instalado. Em vez de comparar o conteúdo dos ficheiros diretamente pode ser comparada o seu *hash* / resumo. O hash ou resumo de um ficheiro é obtido aplicando uma função  $F$  à sequência de bytes  $S$  do ficheiro que produz uma sequência de bytes  $H$  sempre do mesmo tamanho.  $f$  tem a propriedade que, sabendo  $S$  e  $H$ , é muito difícil obter uma sequência  $S'$  tal que  $F(S') == H$ . Um exemplo da função  $F$  é *sha1*; consultar a Wikipedia para ver detalhes sobre o algoritmo *sha-1*.

O comando do Linux *sha1sum <nome ficheiro>* calcula um hash de 20 bytes (160 bits) do ficheiro *<nome ficheiro>*. O screenshot seguinte ilustra o uso do comando *sha1sum*.



```
estudante@estudante-VirtualBox: ~/Desktop
estudante@estudante-VirtualBox:~/Desktop$ sha1sum /etc/passwd
1076d243dc5f2e54669e5969d9708edc6f7ce6d2 /etc/passwd
estudante@estudante-VirtualBox:~/Desktop$
```

Note que o comando escreve uma linha com o hash do ficheiro e o nome do mesmo, separados por 2 espaços.

O trabalho prático envolve fazer um script *bash* que verifica os hashes dos ficheiros contidos numa diretoria com um ficheiro de texto com os hash genuínos

O script recebe dois argumentos:

- O nome de uma diretoria *dir*, contendo várias subdiretorias com ficheiros lá dentro.
- O nome de um ficheiro de texto que contém uma linha para cada um dos ficheiros contidos na diretoria *dir* no formato produzido pelo comando *sha1sum*.

O script deve escrever no terminal:

- OK, se todos os *hashs* dos ficheiros na diretoria que é o 1º argumento são iguais aos que aparecem no ficheiro que é o 2º argumento
- No caso de haver diferenças, o que é escrito no terminal é apenas o output do programa compara, tal como descrito acima.

Para a implementação são válidas as seguintes indicações:

1. O script shell deve usar o comando *find* para produzir um ficheiro de texto em que cada linha contém o output produzido pelo comando *sha1sum*.
2. A comparação dos hashes da diretoria com os hashes genuínos deve usar o programa *compara* desenvolvido na 1ª parte da aula.
3. O ficheiro que é o 2º argumento está ordenado pelo nome do ficheiro. Isto significa que o ficheiro produzido em 1 também tem que o estar. Para essa ordenação, use o comando *sort*. Assuma que os dois ficheiros têm o mesmo número de linhas.

Pode encontrar no CLIP em *Documentação de Apoio -> Problemas* os ficheiros *teste.tar* com diretorias e ficheiros e o ficheiro *hashs\_corretos* que tem os hash dos ficheiros em *teste.tar*. Para as suas experiências faça o seguinte:

- Descarregue os ficheiros *teste.tar* e *hashs\_corretos* para a sua diretoria de trabalho.
- Expanda o ficheiro *teste.tar* com o seguinte comando: *tar xvf teste.tar*
- Invoque o seu script com os argumentos *teste* e *hash\_corretos*. Não deverá ser detetado nenhum hash diferente.
- Experimente agora alterar alguns ficheiros numa das subdiretorias de *teste*, e execute novamente o seu script para verificar quais os ficheiros que foram alterados e que por isso têm hashes diferentes dos que estão no ficheiro *hash\_corretos*.

**Forma de entrega: a divulgar oportunamente através de mensagem enviado via CLIP.**