

## Arquivos de texto e codificação da informação

### Texto Puro (*Plain Text*)

Texto puro, ou *plain text*, é uma sequência pura de *bytes* que representam caracteres um a um. Caractere, por sua vez, é a menor unidade de código que representa um símbolo gráfico (como as letras e números) ou um comando de controle (como tabulação, espaço, fim da linha, entre outros). Dessa forma, um arquivo de texto puro é um arquivo que contém apenas *bytes* que representam símbolos gráficos ou comando de controle.

O uso do texto puro traz a vantagem da independência de programas específicos para sua manipulação pois é, em geral, lido por diversos editores de texto. Além disso é utilizado no desenvolvimento de programas computacionais na maior parte das linguagens de programação existentes, incluindo o *python*.

Outra vantagem associada ao texto puro está ligada ao fato de os *bytes* (ou caracteres) serem independentes entre si. Ainda que um arquivo de texto puro possa conter inúmeros caracteres em seu conteúdo, não há uma relação entre eles. Isso é relevante pois à medida que um arquivo de texto puro sofre uma compressão ou descompressão, ou é copiado, ou é enviado pela rede, ele pode sofrer uma corrupção em seus *bytes*. Contudo, se um *byte* desse arquivo for corrompido não haverá, necessariamente, a corrupção de outros. Esses eventos são independentes e relativos a cada *byte*.

Como exemplo imagine a palavra “Constituição”. Se uma letra qualquer for corrompida e, por isso, transformada em outra, as demais continuam como deveriam ser, como no caso a seguir em que a letra C foi corrompida em K,

ficando “Konstituição”. Ou se as duas letras t forem corrompidas em z, ficando “Conszizuição”. As possibilidades de corrompimento de dados são inúmeras, porém como exemplificado essa é a vantagem do texto puro: o corrompimento de um *byte* não influencia nos demais *bytes*.

### **Texto Rico (*Rich Text*)**

Em contrapartida ao arquivo de texto puro há o texto estilizado (*styled text*), ou o texto rico (*rich text*), que é uma representação de um texto puro adicionada de informações como tamanho da fonte, estilo da fonte, cor, *hipertext link*, espaçamento entre linhas, expressos por *bytes* adicionais. Este tipo de texto é muito utilizado em processadores de texto como o *Microsoft Word*, *LibreOffice Writer* para produzir textos formatados e estilizados de acordo com as preferências do usuário. Como dito anteriormente, neste trabalho não serão utilizados como objetos de estudo arquivos de texto rico.

### **Codificação ASCII**

A codificação ASCII (ou *ascii*), que em inglês significa *American Standard Code for Information Interchange*, é uma correspondência entre *bytes* e caracteres. Pelo fato de ser uma codificação de texto de padrão americano, os símbolos que ela codifica são os elementos do alfabeto inglês e os elementos numerais arábicos. Dessa forma, a codificação ASCII representa, inicialmente, 52 símbolos da língua inglesa (26 letras minúsculas e 26 letras maiúsculas) e mais 10 símbolos numéricos (números de 0 a 9). Além desses símbolos, o código ASCII também representa a pontuação existente na

gramática inglesa como o ponto final, o ponto de exclamação, o ponto de interrogação, a vírgula, entre outros. Esses símbolos citados representam os caracteres visíveis presentes no idioma inglês. Todavia, o ASCII também possui em sua lista de representação caracteres não visíveis, isto é, caracteres que servem para controlar dispositivos que fazem uso da codificação ASCII, como impressoras, leitores de código ASCII, etc. Como exemplo desses caracteres invisíveis estão o espaço (inserido pela barra de espaço do teclado), a tabulação (inserida pela tecla *tab*) e a nova linha (inserida pela tecla *enter*).

Originalmente, o ASCII possui 128 caracteres codificados e para isso utiliza 1 *byte* como uma combinação de 7 dígitos binários (*bits*), indo de 0000 0000 a 0111 1111. O oitavo *bit* (o primeiro à esquerda) foi introduzido, inicialmente, para ser um *bit* de paridade, isto é, um *bit* utilizado apenas para verificar se há erro nos demais. Posteriormente, a codificação ASCII foi estendida para mais 128 caracteres e, com isso, seu *bit* de paridade passou a ser utilizado como oitavo *bit* de codificação transformando o *byte* de 7 *bits* em um *byte* de 8 *bits*, indo de 0000 0000 a 1111 1111, um total de 256 caracteres possíveis.

A imagem a seguir é uma amostra contendo os 6 dos 128 caracteres possíveis da tabela ASCII. A tabela completa contendo a codificação de 7 *bits* e a tabela estendida, de 8 *bits*, estão disponíveis no link <http://www.asciitable.com/>.

Decimal	Binário	ASCII
48	00110000	0
52	00110100	4
63	00111111	?
65	01000001	A
75	01001011	K
123	01111011	{

*Figura 1: Exemplos de codificação ASCII*

A codificação ASCII é, portanto, um padrão possível para a representação de texto puro e contém a simbologia presente na língua inglesa. Já existiram codificações concorrentes que também utilizavam um byte por caractere, mas somente o ASCII tornou-se largamente utilizado. E há, também, outros padrões para representar os diversos alfabetos existentes no mundo, individualmente, e padrões surgindo para unificá-los em apenas uma codificação.

## **Padrão SMTP**

*SMTP (Simple Mail Transfer Protocol)*, em português, Protocolo de Transferência de Correio Simples, é um padrão desenvolvido para troca de mensagens de *email* entre servidores de *email* por meio da internet. Foi inicialmente concebido para que ocorresse a transmissão de mensagens contendo texto puro, codificadas em ASCII apenas. Assim, havia uma limitação do tipo de informação que poderia ser enviada, à época, entre os servidores de email, pois apenas 128 caracteres diferentes estavam disponíveis. Contudo outros métodos de codificação da informação foram desenvolvidos permitindo

assim o envio de outros tipos de dados e arquivos, como imagens, áudios, programas executáveis, etc. Dentre esses novos métodos de codificação destaca-se a codificação *MIME*.

### **Padrão *MIME***

*MIME* (*Multipurpose Internet Mail Extensions*), que em português significa Extensões para Correio de Internet de Múltiplos Propósitos, é um padrão de internet desenvolvido para estender as funcionalidades do formato de email disponíveis à época. Dentre as diversas normas desenvolvidas pelo padrão *MIME* será destacada aqui a que se refere ao envio de dados fora da codificação *ASCII* via *email*.

A codificação *MIME* dá suporte a dois modos de transmitir dados fora do padrão *ASCII* de 7 *bits*: codificando esses dados utilizando o método *quoted-printable* ou o método *base64*. Ambos consistem em transformar dados que não estão no formato de texto puro *ASCII*, de 7 *bits*, em um texto puro codificado em *ASCII*, de 7 *bits*.

A codificação *quoted-printable* é usada para codificar especificamente textos que não estão codificados no formato *ASCII*, como por exemplo textos de outros idiomas, cujos caracteres precisam mais do que 7 *bits* para serem representados. A *quoted-printable* reescreve o texto original substituindo os caracteres que não fazem parte da codificação *ASCII* por caracteres que fazem parte, seguindo uma determinada regra lógica.

Por exemplo, na mensagem “João comprou pão na padaria”, todos os caracteres da mensagem fazem parte da codificação *ASCII*, exceto o caractere

“ã”. Neste caso, a codificação *quoted-printable* possui uma correspondência em ASCII para o caractere “ã” que é o conjunto de 3 caracteres “=E3”. Dessa forma, a mensagem “João comprou pão na padaria” codificada pelo método *quoted-printable* ficaria “Jo=E3o comprou p=E3o na padaria”. Para decodificá-la basta realizar o processo inverso substituindo os caracteres “=E3” pelo caractere “ã”.

O segundo método de codificação proposto pelo *MIME* é a codificação *base64*. Esta codificação é mais genérica que a *quoted-printable* pois é capaz de codificar todo e qualquer arquivo binário em um arquivo de texto puro, seja arquivo de imagem, de áudio, de vídeo, executável, *pdf*, texto. Todos esses formatos de arquivo são, em essência, arquivos binários, isto é, compostos primariamente por dígitos 0 ou 1. Com isso o método *base64* transforma esses arquivos binários em texto utilizando os 62 caracteres da codificação ASCII que são os alfanuméricos (letras e números) mais os três caracteres “/”, “+” e “=”.

Dessa forma é possível transformar um arquivo de áudio, por exemplo, em uma sequência de texto puro.

Como exemplo será feita a conversão da palavra “lei” em ASCII, para a *base64*. Para isso é preciso identificar, para cada caractere da palavra original, o *byte* correspondente. Na codificação ASCII cada caractere tem 8 *bits* (1 *byte*) e na *base64* cada caractere possui 6 *bits*. Portanto, tendo a palavra “lei”, em ASCII, três *bytes*, na codificação *base64* haverá 4 caracteres de 6 *bits*.

A tabela a seguir mostra, de forma enxuta, os passos de conversão:

ASCII	L								e								i							
Decimal	76								101								105							
Binário	0	1	0	0	1	1	0	0	0	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1
Índice	19								6								21							
Base64	T								G								V							

*Figura 2: Conversão para base64*

A linha *ASCII* contém o caractere de 1 *byte* a ser convertido. A linha *Decimal* contém o valor desse *byte* em Decimal. A linha *Binário* contém o caractere em binário de 8 bits (1 *byte*) e 6 bits. A linha *Índice* contém a conversão do binário de 6 *bits* para decimal e, finalmente, a linha *base64* contém o caractere cujo índice da linha acima esteja na tabela de caracteres da *base64*.

Abaixo, segue a tabela de caracteres de conversão para *base64*. A linha índice da tabela acima deve ser procurada nesta segunda tabela.

Índice	Char	Índice	Char	Índice	Char	Índice	Char
0	A	16	Q	31	g	47	w
1	B	17	R	32	h	48	x
2	C	18	S	33	i	49	y
3	D	19	T	34	j	50	z
4	E	20	U	35	k	51	0
5	F	21	V	36	l	52	1
6	G	22	W	37	m	53	2
7	H	23	X	38	n	54	3
8	I	24	Y	39	o	55	4
9	J	25	Z	40	p	56	5
10	K	26	a	41	q	57	6
11	L	27	b	42	r	58	7
12	M	28	c	43	s	59	8
13	N	29	d	44	t	60	9
14	O	30	e	45	u	61	+
15	P	31	f	46	v	62	/

*Figura 3: Tabela de caracteres por base64*

Com isso, a palavra “lei” foi convertida para “TGVq”, na codificação *base64*.

### **Arquivos Mbox (Mailbox)**

*Mailbox* ou *mbox* é um formato de arquivo que armazena, em texto puro, informações concatenadas sobre um ou mais *e-mails* enviados de um servidor a outro.

Um arquivo nesse formato, por conter apenas texto puro, precisa demarcar pontos importantes em seu conteúdo para diferenciar os diversos campos dos *emails* armazenados, como origem, destinatário, assunto, corpo do *email*, anexos e, também, para diferenciar o fim de um *email* e o início de



outro. Dessa forma, o principal requisito utilizado pelo formato *mbox* para o armazenamento de e-mails é que todo *email* deve ser iniciado com uma linha “*From*” (uma linha que comece com os caracteres F, r, o, m), possuir uma série de outras linhas intermediárias contendo outros dados sobre o email e terminar, finalmente, com uma linha totalmente em branco (inclusive sem espaço e sem tabulação). Tudo que estiver compreendido entre a linha “*From*” e a linha em branco corresponderá ao mesmo email.

Para a leitura de um arquivo *mbox* um leitor específico para esse formato de arquivo inicia o processo procurando por linhas “*From*”. Quando uma linha marcada com “*From*” é encontrada o leitor compreende que a mensagem se inicia ali e, então, continua a leitura linha a linha, extraíndo os dados da mensagem, até que a próxima linha “*From*” ou um marcador de fim de arquivo (*End of File*) sejam encontrados. A seguir, um exemplo de uma estrutura de email escrita no formato *mbox*:

```
01. From: Example <example@example.com>
02. MIME-Version: 1.0
03. Content-Type: multipart/mixed;
04.         boundary="XXXXboundary text"
05. This is a multipart message in MIME format.
06. --XXXXboundary text
07. Content-Type: text/plain
08. this is the body text
09. --XXXXboundary text
10. Content-Type: image/jpeg; name="casa.jpg"
11. Content-Disposition: attachment;
```

```

12.          filename="casa.jpg"

13./9j/4AAQSkZJRgABAQAAQABAAD/2wCEAAkGBxMTEhUTEhMVFhU
XFhcXFxUYFhYXGBcVFxgWGBcYFhgYHSggGRolHRcXIjEhJSkrLi4uF
x8zODMsNygtLisBCgoKDg0OGhAQGY8lICUtLS0tLTUtLS0tLS0tLS0
tLS0tLS0wLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLf/AABEIA
LEB...

14. --XXXXboundary text--

```

Este exemplo foi retirado da caixa de *emails* pessoais do autor da pesquisa e sofreu modificações para poder ser apresentado de forma enxuta e com informações genéricas.

Algumas informações relevantes sobre o email de exemplo acima:

- Todo seu conteúdo está escrito em texto puro codificado segundo o padrão ASCII, exigido pelo padrão SMTP;
- A linha 10 informa que há um conteúdo de imagem no email cuja extensão é *jpeg*;
- A linha 11 informa que esse conteúdo de imagem está disponível como anexo;
- A linha 13 dispõe a imagem em anexo sob a forma de texto puro, após passar por uma codificação pelo método *base64* explicado no tópico anterior;

Dessa forma, então, é que os anexos dos mais variados formatos de arquivos são inseridos e enviados via *email*, obedecendo ao padrão SMTP.

## Project Gutenberg

O *Project Gutenberg* é um site fundado por Michael Hart para disponibilizar livros eletrônicos de diversas áreas de forma gratuita. Com isso, um livro eletrônico extraído do site será utilizado para exemplificar, mais à frente, algumas funcionalidades do Linux e das ferramentas de compressão.

O livro a ser usado será o livro *Os Lusíadas*, de Luís Vaz de Camões, que é um arquivo de texto puro (arquivo de extensão .txt) e que será obtido através do link <http://www.gutenberg.org/cache/epub/3333/pg3333.txt>. Será salvo com o nome “os\_lusiadas.txt”.

## Execução de programas em *Linux* e *Windows* via comandos

### Comandos para *Linux*

Uma das formas de se utilizar um sistema operacional é por meio de interfaces gráficas interativas, como as que foram largamente popularizadas pelo *Windows* ao longo dos anos. Tecnicamente, essas interfaces gráficas traduzem de forma intuitiva a comunicação entre o usuário e a máquina. Contudo, há por trás dessa maneira visualmente intuitiva de usar um computador uma série de comandos sendo disparados com ordens de processamento. Dessa forma, uma outra maneira de realizar operações em um computador é por meio de comandos digitados em um interpretador de comandos que, no Linux, é conhecido como *Shell*. A cada comando digitado o *Shell* o interpreta, verifica se há erros na sua estrutura e, caso não haja, o executa.

Para ter acesso ao *Shell* do *Linux* basta abrir o Terminal de comandos, que já foi mencionado em tópicos anteriores. Ao abrir o Terminal, o usuário irá se deparar com uma janela semelhante à seguinte:



*Figura 4: Prompt de comando do Linux*

Note que haverá, geralmente, na margem esquerda, a estrutura <primeiro\_nome>@<segundo nome>. O primeiro nome, que vem antes do arroba é o nome do usuário que está logado no sistema operacional e o segundo nome é o nome da máquina que o usuário está utilizando. Em seguida, há o símbolo \$, que significa que o usuário da máquina está utilizando o terminal de comandos como usuário comum, sem privilégios de administrador. É importante, ainda, salientar que essa estrutura pode ser alterada pelo usuário e acabar adquirindo um outro formato, embora a estrutura padrão explicada acima atenda perfeitamente às necessidades deste trabalho.

Um outro termo que é amplamente utilizado para designar esse terminal de comandos é *prompt de comandos* ou somente *prompt*. Os comandos a serem digitados no *prompt* sempre virão após o símbolo \$.

Com o *prompt* é possível realizar diversas operações que o usuário leigo está acostumado a fazer por meio das interfaces gráficas. É possível navegar pelo sistema de pastas, ou seja, entrar ou sair em um determinado diretório;

copiar, colar e apagar arquivos; modificar arquivos; entre outras inúmeras operações possíveis.

Com isso, serão explicados a seguir alguns comandos importantes para a execução deste trabalho.

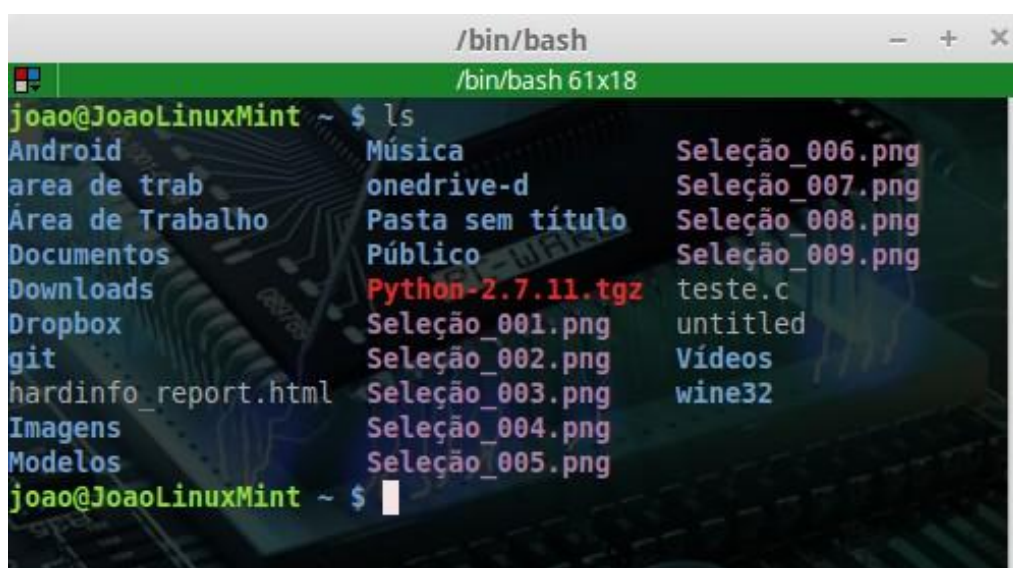
### Diretório padrão (~)

O diretório padrão do Linux é designado pelo símbolo ~, logo após o nome do usuário e da máquina. O Linux reconhece esse diretório como sendo o diretório /home/primeiro\_nome, ou seja, é o diretório padrão do usuário logado no sistema.

### Listar conteúdo de um diretório (ls)

Para listar o conteúdo de um diretório use o comando ls. Este comando, sem opções, exibe o conteúdo do diretório em forma de lista, sem detalhes.

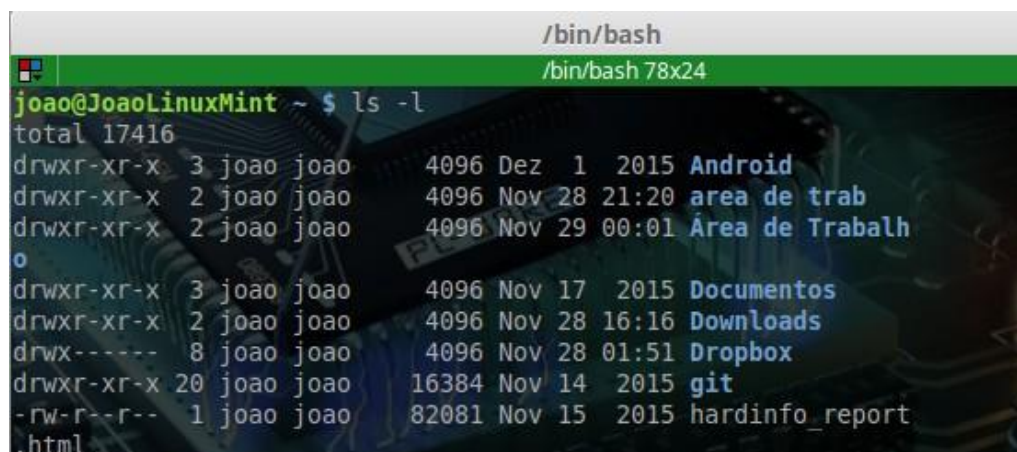
No exemplo a seguir, será listado todo o conteúdo do diretório padrão do computador utilizado no desenvolvimento deste trabalho:



```
/bin/bash
/bin/bash 61x18
joao@JoaoLinuxMint ~ $ ls
Android          Música           Seleção_006.png
area de trab     onedrive-d       Seleção_007.png
Área de Trabalho Pasta sem título Seleção_008.png
Documentos       Público          Seleção_009.png
Downloads        Python-2.7.11.tgz teste.c
Dropbox          Seleção_001.png  untitled
git              Seleção_002.png Vídeos
hardinfo_report.html Seleção_003.png wine32
Imagens          Seleção_004.png
Modelos          Seleção_005.png
joao@JoaoLinuxMint ~ $
```

Figura 5: Comando ls

Mas o comando `ls` também pode ter variantes, como por exemplo o comando `ls -l`, que exibe o conteúdo do diretório com detalhes de tamanho, data de criação, entre outras informações, conforme o exemplo a seguir:



```

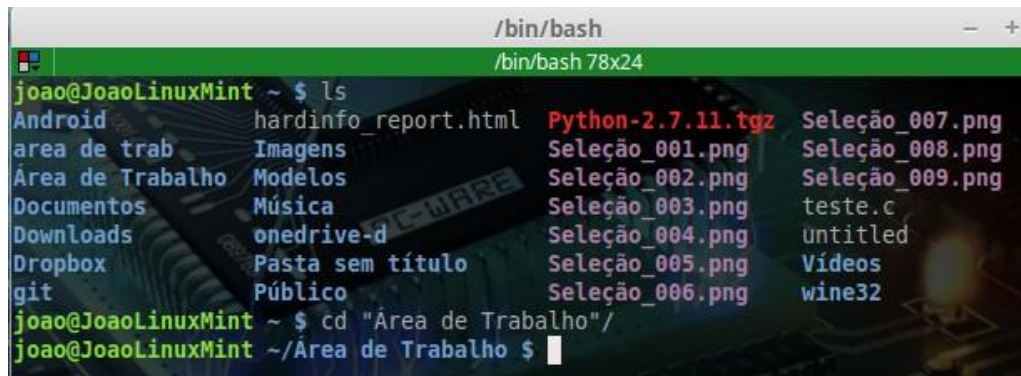
/bin/bash
/bin/bash 78x24
joao@JoaoLinuxMint ~ $ ls -l
total 17416
drwxr-xr-x  3 joao joao    4096 Dez  1  2015 Android
drwxr-xr-x  2 joao joao    4096 Nov 28 21:20 area de trab
drwxr-xr-x  2 joao joao    4096 Nov 29 00:01 Área de Trabalh
o
drwxr-xr-x  3 joao joao    4096 Nov 17  2015 Documentos
drwxr-xr-x  2 joao joao    4096 Nov 28 16:16 Downloads
drwx----- 8 joao joao    4096 Nov 28 01:51 Dropbox
drwxr-xr-x 20 joao joao   16384 Nov 14  2015 git
-rw-r--r--  1 joao joao   82081 Nov 15  2015 hardinfo_report
.html
  
```

Figura 6: Comando `ls -l`

## Navegar por diretórios (`cd`)

O comando para navegar por diretórios é o `cd`, do inglês *change directory*, isto é, “mudar diretório”. Com esse comando é possível mudar de diretório passando por cada nível hierárquico ou indo diretamente ao diretório desejado.

No exemplo a seguir será listado o conteúdo do diretório padrão (`~`) e, em seguida, selecionado algum subdiretório pertencente ao diretório corrente usando o comando `cd <nome do diretório desejado>` ou `cd <nome do diretório desejado>/`. Observe que a diferença entre os dois formatos do comando `cd` exemplificados acima está no uso ou não da barra “/” ao final do nome do diretório desejado. Esta barra ao final é opcional, não interferindo assim na execução do comando:



```

/bin/bash
/bin/bash 78x24
joao@JoaoLinuxMint ~ $ ls
Android          hardinfo_report.html  Python-2.7.11.tgz  Seleção_007.png
area de trab     Imagens               Seleção_001.png    Seleção_008.png
Área de Trabalho Modelos               Seleção_002.png    Seleção_009.png
Documentos       Música                Seleção_003.png    teste.c
Downloads        onedrive-d            Seleção_004.png    untitled
Dropbox          Pasta sem título      Seleção_005.png    Vídeos
git              Público               Seleção_006.png    wine32
joao@JoaoLinuxMint ~ $ cd "Área de Trabalho"/
joao@JoaoLinuxMint ~/Área de Trabalho $

```

*Figura 7: Comando cd*

No caso exemplificado foi utilizado o comando `cd` para entrar no diretório Área de Trabalho. Perceba que após o comando ter sido executado, o diretório corrente não é mais o diretório padrão, mas sim o diretório `~/Área de Trabalho`. Esse processo é análogo, quando se utiliza a interface gráfica, a dar duplo clique sobre a pasta Área de Trabalho.

Para retornar para um diretório anterior, basta digitar “`cd ..`” (`cd` seguido de um espaço simples e de dois pontos finais). No exemplo anterior, para retornar ao diretório padrão, a partir do qual entramos no diretório Área de Trabalho, o comando seria o seguinte:



```

/bin/bash
/bin/bash 78x24
joao@JoaoLinuxMint ~/Área de Trabalho $ cd ..
joao@JoaoLinuxMint ~ $

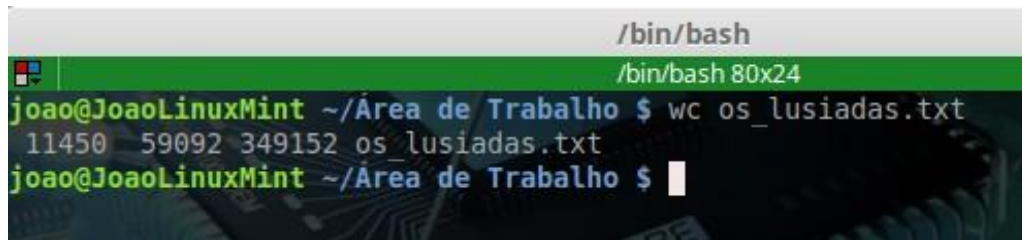
```

*Figura 8: Comando cd ..*

O comando `cd` pode ser usado repetidas vezes, tanto para descer a subdiretórios quanto para retornar a níveis superiores, além de poder ser usado para acessar um diretório qualquer desde que o caminho integral do diretório seja passado e que comece com o caractere “/”.

## Contar número de palavras (wc)

O comando `wc`, do inglês *word count*, que em português significa contar palavras faz exatamente o que o nome sugere, além de contar linhas e caracteres de um arquivo. Usando o arquivo “`os_lusiadas.txt`”, vamos contar quantas linhas, palavras e caracteres há no arquivo:



```
/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ wc os_lusiadas.txt
11450  59092 349152 os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $
```

Figura 9: Comando `wc`

O resultado apresentado pelo comando obedece à seguinte ordem: número de linhas, número de palavras e número de caracteres.

Há variações do comando para apresentar apenas um resultado específico:

`wc -l “<nome do arquivo>”` → mostra apenas o total de linhas

`wc -m “<nome do arquivo>”` → mostra apenas o total de caracteres

`wc -w “<nome do arquivo>”` → mostra apenas o total de palavras

## Exibir informações de estado de um arquivo (stat)

O comando `stat` e suas variações fornecem informações sobre um determinado arquivo. Este comando é útil, por exemplo, para se obter o tamanho do arquivo. Para isso, o comando é `stat -c %s “<nome do arquivo>”`, como no exemplo a seguir:





```
/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ stat -c %s os_lusiadas.txt
349152
joao@JoaoLinuxMint ~/Área de Trabalho $
```

Figura 10: Comando *stat*

No exemplo acima, o comando digitado exibiu o tamanho do arquivo, em bytes.

### Exibir o tempo de execução de um processo (*time*)

Quando um comando, um script ou um processo é executado, naturalmente é levado algum tempo até que a execução termine. O comando *time* exhibe ao usuário três tempos de execução distintos: tempo *real*, tempo *user* e tempo *sys*. O tempo *real* é o tempo total da operação, desde sua inicialização até sua finalização, isto é, a hora final menos a hora inicial. O tempo *user* é o tempo usado pelo processo (programa em execução) apenas em procedimentos acessíveis pelo usuário logado, e o tempo *sys* é o tempo usado pelo processo em procedimentos acessíveis apenas pelo *kernel* (manipulação de dispositivos, acesso à memória, etc).

Para análise dos resultados deste trabalho o tempo de interesse é a soma entre os tempos "*user*" e "*sys*". Esse resultado pode ser menor que o tempo "*real*", quando há um único processador que está rodando vários processos ao mesmo tempo (multitarefa), ou maior, quando há vários processadores ("*dual core*", "*quad core*", "*octocore*" etc.) que podem trabalhar simultaneamente no mesmo processo e ter seus consumos somados entre si. O comando para se obter o tempo de execução é *time* "<nome do comando,

*script ou processo>*”, como no exemplo abaixo, que exibe os tempos de execução do comando listar conteúdo do diretório (*ls*):

```

/bin/bash
/bin/bash 78x24
joao@JoaoLinuxMint ~ $ time ls
Android          hardinfo_report.html Python-2.7.11.tgz Seleção_007.png
area de trab     Imagens          Seleção_001.png  Seleção_008.png
Area de Trabalho Modelos          Seleção_002.png  Seleção_009.png
Documentos       Música           Seleção_003.png  teste.c
Downloads        onedrive-d       Seleção_004.png  untitled
Dropbox          Pasta sem título Seleção_005.png  Videos
git              Público          Seleção_006.png  wine32

real    0m0.033s
user    0m0.000s
sys     0m0.004s
joao@JoaoLinuxMint ~ $

```

Figura 11: Comando *time*

## Comparar arquivos (*cmp*)

O comando *cmp* é usado para comparar dois arquivos entre si, em especial arquivos binários. Caso os arquivos sejam diferentes, o comando exibirá o byte e o número da linha a que pertence a diferença entre os arquivos. Se os arquivos forem idênticos, por padrão o comando *cmp* não exibirá nenhum resultado. O comando possui a seguinte estrutura: *cmp* “<arquivo1>” “<arquivo2>”.

## Diferenciar arquivos (*diff*)

Um modo mais sofisticado de comparar arquivos de texto, especialmente códigos de programas e “scripts”, é utilizando o comando *diff*.

O comando *diff* exibe as diferenças entre arquivos distintos. Seu código possui a seguinte estrutura: *diff* <opções> “<arquivo1>” “<arquivo2>”.

Algumas opções possíveis para variar este comando são:

-B → para ignorar linhas em branco

-q → para mostrar apenas se os arquivos 1 e 2 são iguais ou não

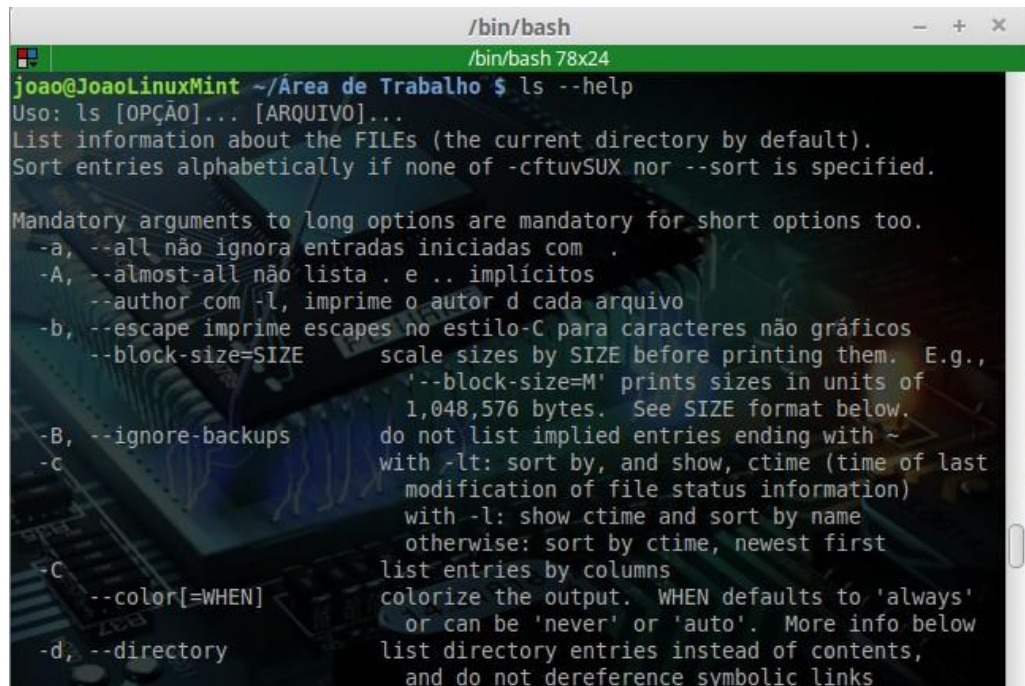
-i → para ignorar letras maiúsculas ou minúsculas

### **Comparar arquivos de texto com dados ordenados (comm)**

O comando *comm* é usado para comparar dois arquivos de texto puro cujos dados estejam ordenados. A comparação realizada resulta em 3 colunas: a primeira coluna contém os dados que aparecem apenas no arquivo 1; a segunda coluna contém os dados que aparecem somente no arquivo 2; e a terceira coluna contém os dados comuns a ambos arquivos.

### **Detalhar informações sobre um comando (help)**

Por padrão, em todo ambiente Linux o usuário conta com uma documentação sobre cada comando disponível. Para acessar a documentação de ajuda de um comando qualquer basta digitar o seguinte comando no prompt: *<nome do comando> --help*, como no exemplo a seguir:



```

/bin/bash
/bin/bash 78x24
joao@JoaolinuxMint ~/Área de Trabalho $ ls --help
Uso: ls [OPÇÃO]... [ARQUIVO]...
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all não ignora entradas iniciadas com .
-A, --almost-all não lista . e .. implícitos
--author com -l, imprime o autor d cada arquivo
-b, --escape imprime escapes no estilo-C para caracteres não gráficos
--block-size=SIZE scale sizes by SIZE before printing them. E.g.,
--block-size=M prints sizes in units of
1,048,576 bytes. See SIZE format below.
-B, --ignore-backups do not list implied entries ending with ~
-c with -lt: sort by, and show, ctime (time of last
modification of file status information)
with -l: show ctime and sort by name
otherwise: sort by ctime, newest first
-C list entries by columns
--color[=WHEN] colorize the output. WHEN defaults to 'always'
or can be 'never' or 'auto'. More info below
-d, --directory list directory entries instead of contents,
and do not dereference symbolic links

```

Figura 12: Comando help

O comando *help* exibe informações detalhadas de como funciona o comando em questão, por isso é essencial que o usuário crie o hábito de usar o *help* para descobrir novas possibilidades além das explicadas neste trabalho.

## Comandos para Windows

Assim como ocorre no *Linux*, o *Windows* também oferece um *prompt* de comando para execução de operações via linha de comandos e a forma de se utilizar o *prompt* do *Windows* é similar à do *Linux*.

Para ter acesso ao *Shell* (interpretador de comandos) do *Windows* abra o menu iniciar e procure pelo aplicativo *prompt de comando*.

Ao abrir o *Prompt*, o usuário irá se deparar com uma janela semelhante à seguinte:



*Figura 13: Prompt de comando do Windows*

Note que no *prompt* do *Windows* haverá sempre, na margem esquerda, a estrutura `C:\Users\<nome do usuário>`. O nome do usuário corresponde ao usuário que está logado no sistema operacional. Em seguida, há o símbolo `>`, que significa que a partir deste símbolo é o local aonde os comandos serão inseridos.

A seguir, alguns comandos para o ambiente *Windows* serão explicados: *cd*, *dir*, *fc* e *help*.

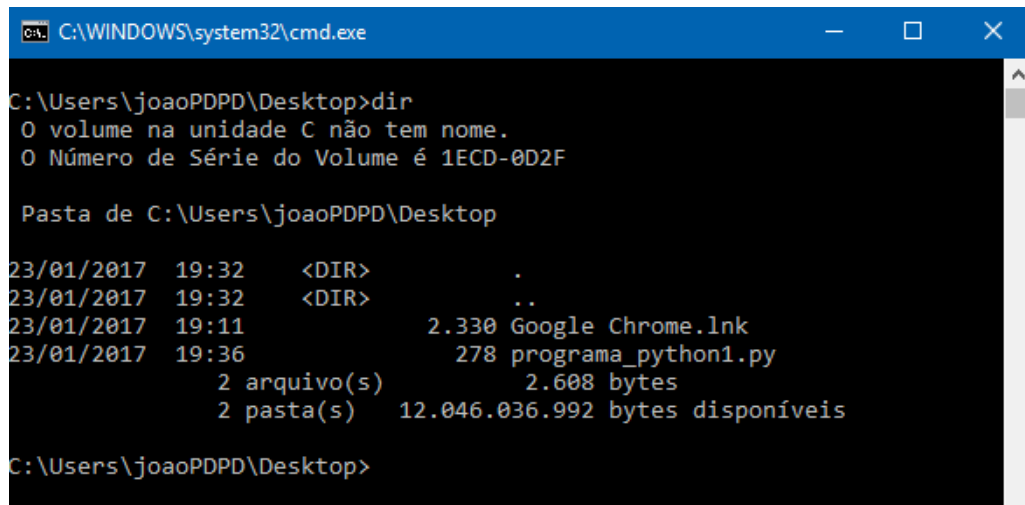
### **Navegar por diretórios (*cd*)**

O comando para navegar por diretórios em ambiente *Windows* possui a mesma estrutura e funcionamento do comando em ambiente Linux.

### **Listar conteúdo de um diretório (*dir*)**

Para listar o conteúdo de um diretório use o comando *dir*. Este comando, sem opções, exibe o conteúdo do diretório em forma de lista contendo data e hora, tamanho em bytes e nome do arquivo ou subdiretório.

No exemplo a seguir, será listado todo o conteúdo do diretório *Desktop*:



```

C:\WINDOWS\system32\cmd.exe

C:\Users\joaoPDPD\Desktop>dir
O volume na unidade C não tem nome.
O Número de Série do Volume é 1ECD-0D2F

Pasta de C:\Users\joaoPDPD\Desktop

23/01/2017  19:32    <DIR>          .
23/01/2017  19:32    <DIR>          ..
23/01/2017  19:11                2.330 Google Chrome.lnk
23/01/2017  19:36                278 programa_python1.py
                2 arquivo(s)          2.608 bytes
                2 pasta(s)      12.046.036.992 bytes disponíveis

C:\Users\joaoPDPD\Desktop>

```

Figura 14: Comando *dir*

## Comparar arquivos (fc)

O comando *fc* é usado para comparar arquivos entre si. É similar ao comando *cmp* utilizado no ambiente *Linux*. O comando possui a seguinte estrutura: *fc <opções> “<arquivo1>” “<arquivo2>”*.

Algumas opções para personalizar este comando são:

*/B* → para executar comparação binária

*/C* → para ignorar maiúsculas e minúsculas

*/L* → para comparar arquivos como texto ASCII

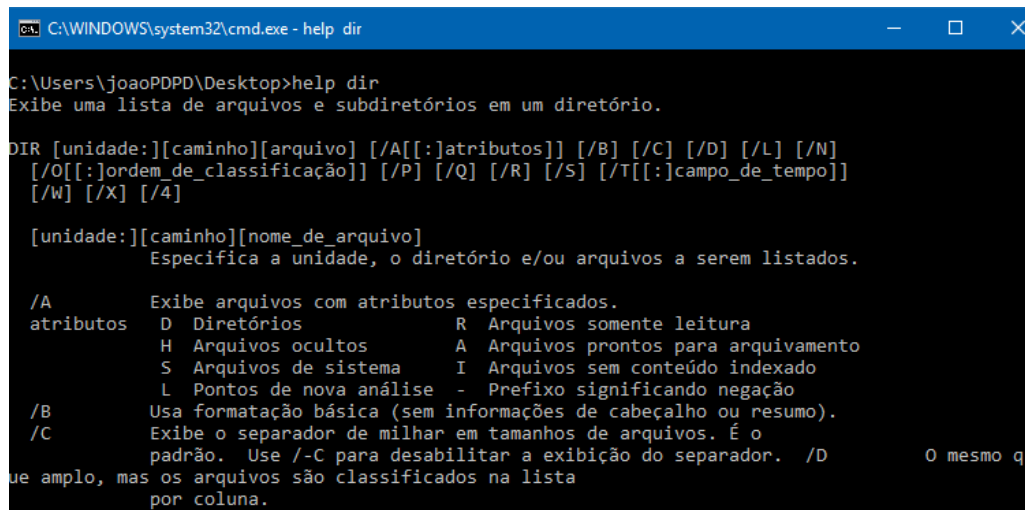
*/N* → para exibir os números de linha em uma comparação ASCII

*/U* → para comparar arquivos como texto Unicode

*/W* → para compactar espaços em branco (tabulações e espaços) e depois comparar

## Detalhar informações sobre um comando (help)

Por padrão, em todo o ambiente *Windows* o usuário conta com uma documentação sobre cada comando disponível. Para acessar a documentação de ajuda de um comando qualquer basta digitar o seguinte comando no prompt: *help <nome do comando>*, como no exemplo a seguir:



```

C:\WINDOWS\system32\cmd.exe - help dir

C:\Users\joaoPDPD\Desktop>help dir
Exibe uma lista de arquivos e subdiretórios em um diretório.

DIR [unidade:][caminho][arquivo] [/A[[:]atributos]] [/B] [/C] [/D] [/L] [/N]
  [/O[[:]ordem_de_classificação]] [/P] [/Q] [/R] [/S] [/T[[:]campo_de_tempo]]
  [/W] [/X] [/4]

[unidade:][caminho][nome_de_arquivo]
  Especifica a unidade, o diretório e/ou arquivos a serem listados.

/A      Exibe arquivos com atributos especificados.
atributos  D Diretórios          R Arquivos somente leitura
           H Arquivos ocultos    A Arquivos prontos para arquivamento
           S Arquivos de sistema I Arquivos sem conteúdo indexado
           L Pontos de nova análise - Prefixo significando negação
/B      Usa formatação básica (sem informações de cabeçalho ou resumo).
/C      Exibe o separador de milhar em tamanhos de arquivos. É o
         padrão. Use /-C para desabilitar a exibição do separador. /D      O mesmo q
         ue amplo, mas os arquivos são classificados na lista
         por coluna.
  
```

Figura 15: Comando help

## Introdução às ferramentas de compressão/descompressão

Algumas ferramentas de compressão e descompressão de dados já estão disponíveis para sistemas Linux. Contudo, algumas destas ferramentas necessitam de instalação enquanto outras já estão incorporadas ao sistema operacional. A seguir serão listadas todas as ferramentas que serão utilizadas ao longo do estudo e, para cada uma delas, serão demonstrados os procedimentos para verificar se estão instaladas e para instalá-las, caso necessário.

Historicamente, o termo *zip* foi criado por Phil Katz para especificar um tipo de formato de arquivo que suportasse compressão de dados sem perdas. As ferramentas de compressão e descompressão de dados costumam, portanto, utilizar o termo *zip* em seus nomes.

A seguir, serão vistas 4 ferramentas comerciais de compressão de dados e que são todas “*open-source*” (livre) e gratuitas. Os algoritmos que essas ferramentas implementam serão estudados com maior profundidade posteriormente em capítulos exclusivamente dedicados a eles.

### ***Gzip (GNU zip)***

O compressor de arquivos *gzip* foi incorporado ao sistema Linux pelo *GNU Project* e, com isso, é uma ferramenta encontrada nativamente nas distribuições Linux. O *gzip* é baseado nos algoritmos de compressão *LZ77* e *Huffman Coding*. Os arquivos compactados pelo *gzip* possuem sempre a extensão *.gz* (não possuem relação com arquivos de extensão *.zip*) e para descompactar arquivos do tipo *.gz* é necessário o descompressor *gunzip*.

### ***Bzip2***

O *Bzip2* utiliza os algoritmos de compressão *Burrows-Wheeler* e *Huffman coding*. Os arquivos compactados pelo *bzip2* possuem a extensão *.bz2* e para descompactar arquivos do tipo *.bz2* é necessário o descompressor *bunzip2*.



## ***XZ Utils***

O *xz utils* foi desenvolvido baseado no algoritmo de compressão de dados LZMA (*Lempel – Ziv – Markov Chain Algorithm*). O *XZ Utils* gera arquivos compactados no formato *.xz* e para descompactá-los é necessário o descompressor *unxz*. Contudo o descompressor *xz* também reconhece e é capaz de descompactar arquivos com a extensão *.lzma*.

Importante salientar que o compressor *LZMA Utils* foi substituído pelo compressor *XZ Utils*.

## ***Lzop***

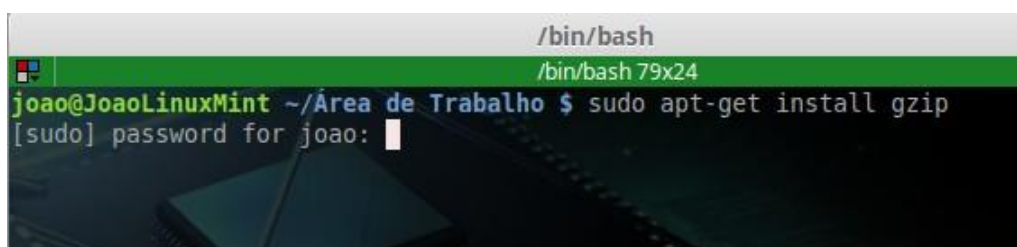
O *Lzop* é um compressor e descompressor que implementa o algoritmo de compressão Lzo (*Lempel-Ziv-Oberhumer*). O *Lzop* foi desenvolvido com o intuito principal de comprimir dados rapidamente, abrindo mão da taxa de compressão. Os arquivos gerados por ele possuem a extensão *.lzo* e o próprio *Lzop* descompacta os arquivos.

Uma observação relevante acerca de todos os compressores demonstrados acima é a possibilidade de executar compressões de dados objetivando uma velocidade de compressão maior em detrimento de uma qualidade de compressão menor, ou vice-versa, quando opta-se pela qualidade da compressão (isto é, maior taxa de compressão) à velocidade.

## Instalação das ferramentas de compressão/descompressão

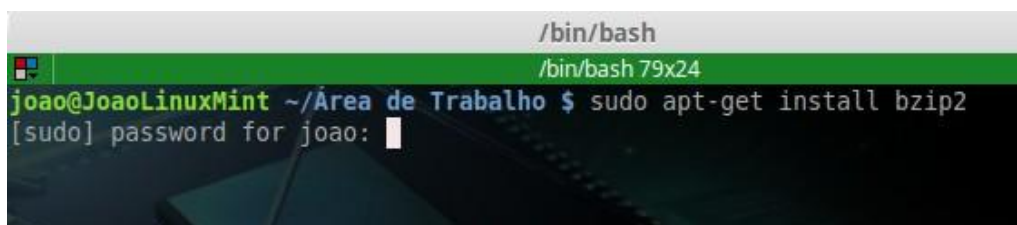
Para utilizar as ferramentas listadas acima é necessário que estejam instaladas no Linux. Para evitar que se trabalhe com versões desatualizadas de cada uma delas será explicado a seguir como que se realiza a instalação. Caso o Linux reconheça que a ferramenta já esteja instalada ele tentará atualizar para a versão mais recente. Se porventura a ferramenta disponível já estiver atualizada, ele então emitirá um aviso dizendo que a versão mais nova já está em uso.

Para instalar as ferramentas de compressão, abra o terminal do Linux e, para cada ferramenta, digite o comando `sudo apt-get install <nome da ferramenta>` e, em seguida, digite a senha do usuário da máquina, conforme imagens a seguir:

A terminal window titled '/bin/bash' with a green header bar. The prompt shows the user 'joao' at 'JoaoLinuxMint' in the directory '~/Área de Trabalho'. The command 'sudo apt-get install gzip' has been entered. The next line shows '[sudo] password for joao:' followed by a white cursor box.

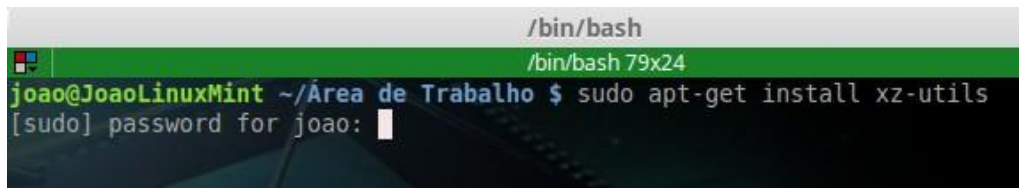
```
/bin/bash
/bin/bash 79x24
joao@JoaoLinuxMint ~/Área de Trabalho $ sudo apt-get install gzip
[sudo] password for joao: 
```

Figura 16: Instalando gzip no Linux

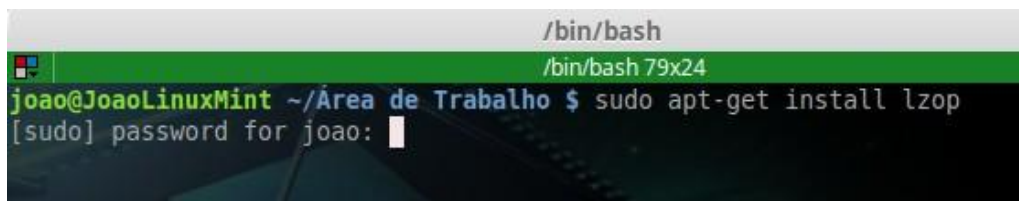
A terminal window titled '/bin/bash' with a green header bar. The prompt shows the user 'joao' at 'JoaoLinuxMint' in the directory '~/Área de Trabalho'. The command 'sudo apt-get install bzip2' has been entered. The next line shows '[sudo] password for joao:' followed by a white cursor box.

```
/bin/bash
/bin/bash 79x24
joao@JoaoLinuxMint ~/Área de Trabalho $ sudo apt-get install bzip2
[sudo] password for joao: 
```

Figura 17: Instalando bzip2 no Linux

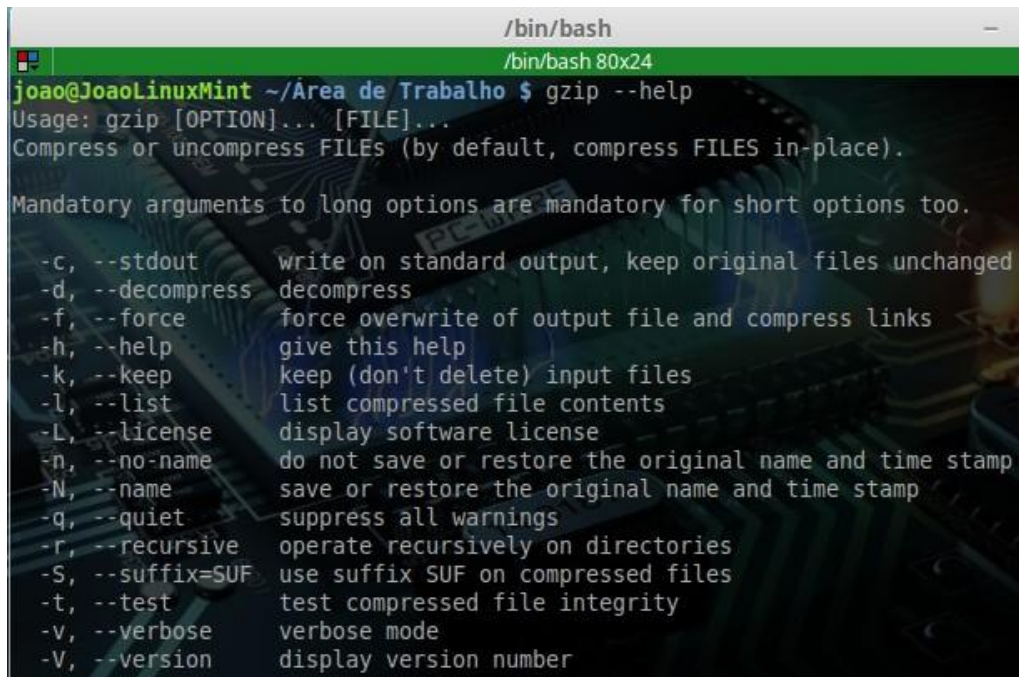
A terminal window titled '/bin/bash' with a green header bar showing '/bin/bash 79x24'. The prompt is 'joao@JoaoLinuxMint ~/Área de Trabalho \$'. The command 'sudo apt-get install xz-utils' has been entered. The next line shows '[sudo] password for joao:' followed by a white cursor.

*Figura 18: Instalando xz-utils no Linux*

A terminal window titled '/bin/bash' with a green header bar showing '/bin/bash 79x24'. The prompt is 'joao@JoaoLinuxMint ~/Área de Trabalho \$'. The command 'sudo apt-get install lzop' has been entered. The next line shows '[sudo] password for joao:' followed by a white cursor.

*Figura 19: Instalando lzop no Linux*

Após a instalação ou atualização das ferramentas de compressão, confirme se realmente estão instaladas. Para isso digite o comando de ajuda de cada ferramenta no terminal. Como no exemplo a seguir:



```

/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ gzip --help
Usage: gzip [OPTION]... [FILE]...
Compress or uncompress FILEs (by default, compress FILEs in-place).

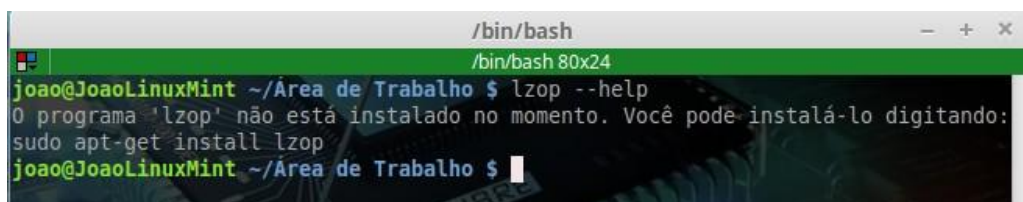
Mandatory arguments to long options are mandatory for short options too.

  -c, --stdout      write on standard output, keep original files unchanged
  -d, --decompress  decompress
  -f, --force       force overwrite of output file and compress links
  -h, --help        give this help
  -k, --keep        keep (don't delete) input files
  -l, --list        list compressed file contents
  -L, --license     display software license
  -n, --no-name     do not save or restore the original name and time stamp
  -N, --name        save or restore the original name and time stamp
  -q, --quiet       suppress all warnings
  -r, --recursive  operate recursively on directories
  -S, --suffix=SUF use suffix SUF on compressed files
  -t, --test        test compressed file integrity
  -v, --verbose     verbose mode
  -V, --version     display version number

```

*Figura 20: Verificando se instalou*

Se o Linux reconhecer o comando digitado, então a ferramenta estará instalada corretamente. Do contrário, o Linux não reconhecerá o comando e emitirá um alerta de comando inválido, como na imagem a seguir:



```

/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ lzop --help
O programa 'lzop' não está instalado no momento. Você pode instalá-lo digitando:
sudo apt-get install lzop
joao@JoaoLinuxMint ~/Área de Trabalho $

```

*Figura 21: Terminal Linux não reconhece um programa*

Se isso ocorrer, retorne ao passo de instalação da ferramenta e realize o procedimento de instalação.

## Comandos básicos das ferramentas de compressão/descompressão

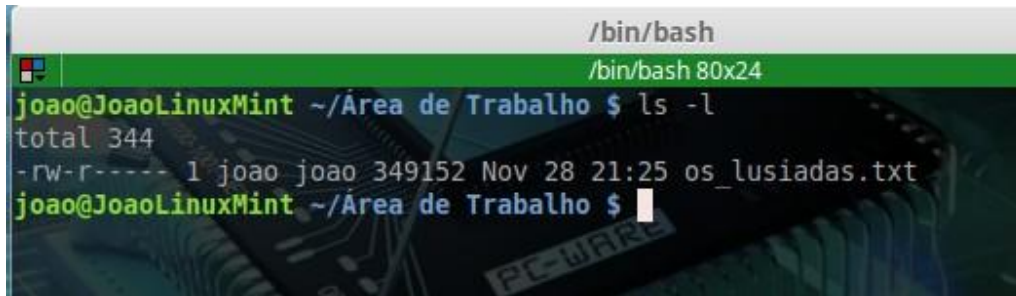
### Compressão de arquivos

Para comprimir arquivos usando as ferramentas acima é necessário, primeiramente, um arquivo a ser compactado. Para uma visualização melhor do poder de compressão das ferramentas, será usado um arquivo de texto puro com um tamanho de arquivo relevante. Contudo, outros tipos de arquivos poderiam ser testados também, como arquivos de imagem, som, vídeo, entre outros.

Como dito anteriormente, neste capítulo o arquivo de texto a ser usado para exemplificação das ferramentas de compressão de dados será o livro *Os Lusíadas*.

Para realizar os testes com as ferramentas de compressão de dados será adotado como diretório de referência o diretório Área de Trabalho, do Linux. Neste diretório ficarão o arquivo de texto original e os arquivos compactados que serão criados. Este diretório tem a vantagem de ser a interface gráfica principal do *Windows* e do *Linux*.

Após salvar o arquivo de texto *Os Lusíadas* na Área de Trabalho, abra o terminal, entre no diretório Área de Trabalho e digite o comando `ls -l`, conforme a imagem a seguir:



```

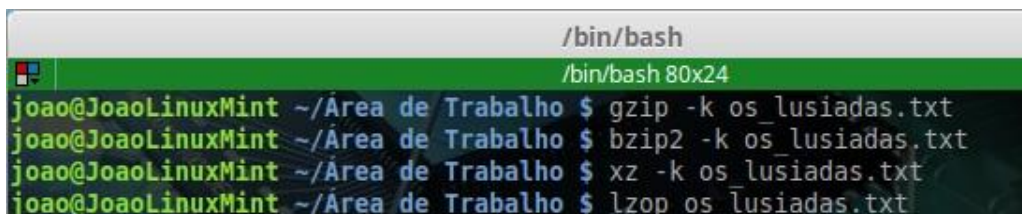
/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls -l
total 344
-rw-r----- 1 joao joao 349152 Nov 28 21:25 os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $

```

*Figura 22: Listando diretório Área de Trabalho*

A imagem acima mostra que na Área de Trabalho há apenas um arquivo chamado “os lusíadas.txt” cujo tamanho é de 349.152 bytes. Este é o arquivo original a ser compactado com as ferramentas previamente selecionadas.

O comando para compactar um arquivo possui a mesma estrutura para todas as ferramentas: *<nome do compressor> “<nome do arquivo a compactar>”*. Contudo, é importante ressaltar que, com exceção do compressor *lzop*, todos os outros (*gzip*, *bzip2* e *xz*) excluem, por padrão, o arquivo original após sua compressão. Por isso, para estes três compressores o comando a ser usado para a compressão do arquivo será *<nome do compressor> -k “<nome do arquivo a compactar>”*. Neste caso, *-k* faz com que o compressor mantenha o arquivo original após sua compactação. Abaixo, segue imagem exemplificando o processo de compactação para cada compressor:



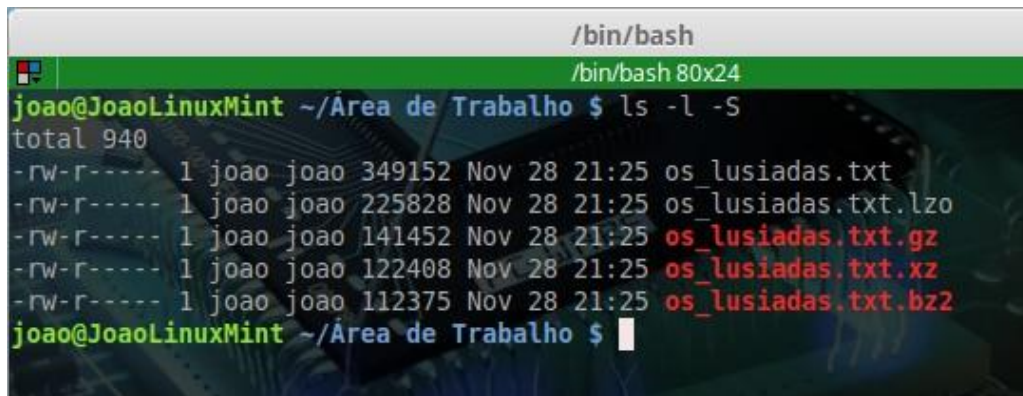
```

/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ gzip -k os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $ bzip2 -k os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $ xz -k os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $ lzop os_lusiadas.txt

```

*Figura 23: Comandos para comprimir arquivo*

Após a compactação utilizando todos as ferramentas de compressão, digite o comando *ls -l -S* para listar todos os arquivos da Área de Trabalho ordenados por tamanho decrescente de arquivo:



```
/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls -l -S
total 940
-rw-r----- 1 joao joao 349152 Nov 28 21:25 os_lusiadas.txt
-rw-r----- 1 joao joao 225828 Nov 28 21:25 os_lusiadas.txt.lzo
-rw-r----- 1 joao joao 141452 Nov 28 21:25 os_lusiadas.txt.gz
-rw-r----- 1 joao joao 122408 Nov 28 21:25 os_lusiadas.txt.xz
-rw-r----- 1 joao joao 112375 Nov 28 21:25 os_lusiadas.txt.bz2
joao@JoaoLinuxMint ~/Área de Trabalho $
```

Figura 24: Listagem diretório com ordem decrescente de tamanho de arquivo

Note, agora, que além do arquivo original “os lusiadas.txt” há mais quatro arquivos com as extensões de cada compactador de arquivos. Estes são os arquivos compactados.

Dessa forma fica demonstrado como compactar um arquivo qualquer usando as ferramentas de compressão escolhidas.

## Descompressão de arquivos

Para desfazer o processo de compressão, isto é, a descompactação dos arquivos, há duas maneiras: usar o respectivo descompressor de dados para o arquivo ou utilizar o próprio compressor sinalizado com a opção de descompressão.

## Descompactar arquivos utilizando o descompressor

A correspondência entre o compressor e seu descompressor é trivial:

Compressor	Descompressor
gzip	gunzip
bzip2	bunzip2
xz	unxz
lzop	(Não possui)

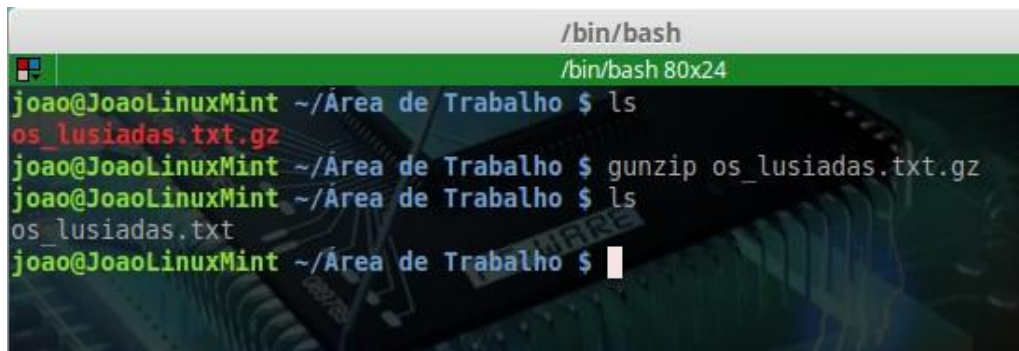
*Figura 25: Ferramentas de compressão e descompressão*

O comando a ser digitado no terminal para descompactar um arquivo é o seguinte: `<nome do descompressor> "<nome do arquivo compactado>"`. Este método gera um arquivo descompactado com o mesmo nome do arquivo original.

Para descompactar para um novo arquivo com nome diferente do arquivo original o comando segue a seguinte estrutura: `<nome do descompressor> -c "<nome do arquivo compactado>" > "<nome do arquivo destino>"`. Este método é útil para manter o arquivo original e o arquivo novo como base de comparação do efeito compressão e descompressão.

A seguir será exemplificada a descompactação de um arquivo por meio de seu descompressor, usando o método para gerar um arquivo com nome igual ao original e um arquivo com nome diferente.



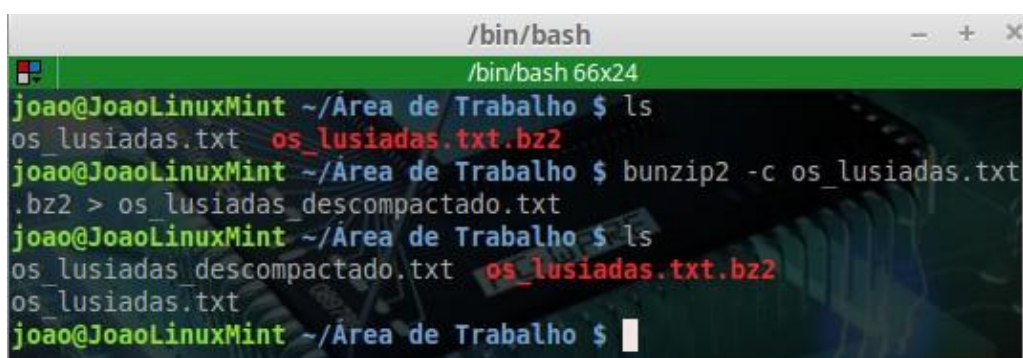


```

/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt.gz
joao@JoaoLinuxMint ~/Área de Trabalho $ gunzip os_lusiadas.txt.gz
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $

```

Figura 26: Descompressão de arquivo .gz mantendo o nome original



```

/bin/bash
/bin/bash 66x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt os_lusiadas.txt.bz2
joao@JoaoLinuxMint ~/Área de Trabalho $ bunzip2 -c os_lusiadas.txt
.bz2 > os_lusiadas_descompactado.txt
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas_descompactado.txt os_lusiadas.txt.bz2
os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $

```

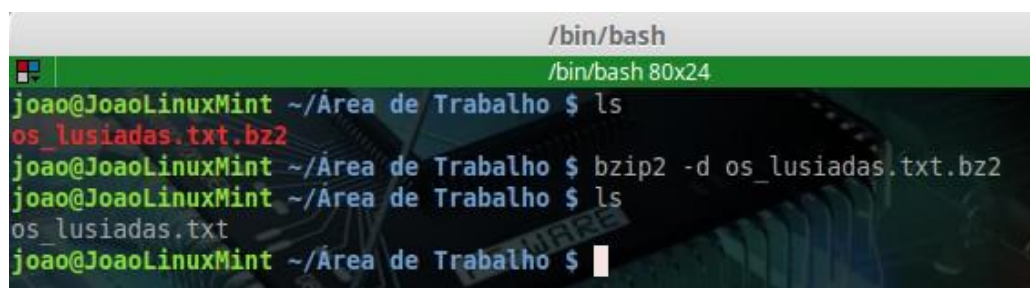
Figura 27: Descompressão de arquivo .bz2 gerando um novo arquivo

## Descompactar utilizando o compressor

O comando para descompactar arquivos utilizando o próprio compressor de dados é semelhante ao comando para compactar os arquivos, com a diferença que há um sinalizador no comando de que o processo é de descompressão. Dessa forma, a estrutura do comando é a seguinte: *<nome do compressor> -d "<nome do arquivo compactado>"*. Note que a única diferença entre este comando e o comando de compressão é a presença do `-d`, que significa *decompression*, isto é, em português significa descompressão. Este método gera um arquivo descompactado com o mesmo nome do arquivo original.

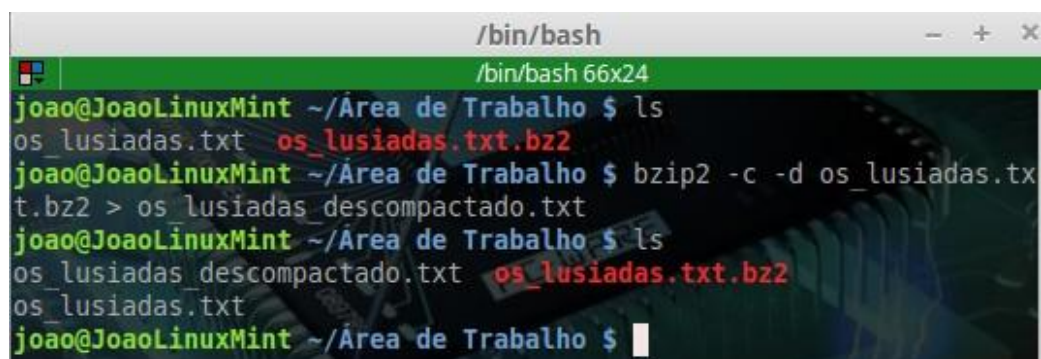
Para descompactar para um novo arquivo com nome diferente do arquivo original o comando segue a seguinte estrutura: <nome do compressor> -c -d “<nome do arquivo compactado>” > “<nome do arquivo destino>”. Este método é útil para manter o arquivo original e o arquivo novo como base de comparação do efeito compressão e descompressão.

Com isso, a descompactação utilizando o compressor é realizada, usando o método para gerar um arquivo com nome igual ao original e um arquivo com nome diferente, da seguinte forma:



```
/bin/bash
/bin/bash 80x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt.bz2
joao@JoaoLinuxMint ~/Área de Trabalho $ bzip2 -d os_lusiadas.txt.bz2
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $
```

Figura 28: Descompressão de arquivo .bz2 mantendo o nome original



```
/bin/bash
/bin/bash 66x24
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas.txt os_lusiadas.txt.bz2
joao@JoaoLinuxMint ~/Área de Trabalho $ bzip2 -c -d os_lusiadas.tx
t.bz2 > os_lusiadas_descompactado.txt
joao@JoaoLinuxMint ~/Área de Trabalho $ ls
os_lusiadas_descompactado.txt os_lusiadas.txt.bz2
os_lusiadas.txt
joao@JoaoLinuxMint ~/Área de Trabalho $
```

Figura 29: Descompressão de arquivo .bz2 gerando um novo arquivo

## Testes primários de compressão

Nesta seção serão apresentados os resultados de taxas e tempos de compactação de 3 arquivos distintos utilizando as ferramentas comerciais explicadas anteriormente (*gzip*, *bzip2* e *xz*). Os arquivos testados serão arquivos de email do professor orientador, arquivos de email da Enron e um arquivo contendo a obra *Os Lusíadas*.

Nos testes de compressão dos arquivos de email do professor orientador e da Enron foi utilizado um sistema *Windows 10 Home*, processador Intel Celeron J1800 *dualcore* a 2,41 GHz e 2,00 GB de memória RAM. O sistema é 64 *bits* e os programas também 64 *bits* compilados para *Windows*: *gzip* 1.6; *bzip2* 1.0.6; *xz* 5.2.2. Já para o arquivo *Os Lusíadas* foi utilizado um sistema *Linux Mint 17.2 Cinnamon 64-bit*, processador Intel Pentium Dual Core T4400 2,20 Ghz e 1,9GB de memória RAM.

## Arquivos mbox cedidos pelo orientador

Apresentamos a seguir os resultados tabelados obtidos com um arquivo *mbox* gentilmente fornecido pelo orientador. Como se trata de um arquivo contendo informações de trabalho confidenciais, não é possível disponibilizá-lo para que o leitor experimente reproduzir as operações em cima do mesmo material. Este caso particular servirá apenas para exibição dos resultados.

A tabela a seguir mostra, para cada ferramenta comercial, o tamanho final dos arquivos e suas respectivas taxas de compactação após a compressão do arquivo *mbox*:

Tamanho do arquivo original: 4.259.498.761 bytes		
Compressor	Tamanho após compactação (bytes)	Taxa de compactação (%)
gzip	2.663.514.597	62,5
bzip2	2.599.790.052	61,0
xz	2.152.207.300	50,5

Figura 30: Dados de compactação do arquivo mbox do orientador

A próxima tabela evidencia cada tempo (*real*, *user* e *sys*) gasto com a operação de compactação do arquivo *mbox* do orientador:

Apuração dos tempos de compactação do arquivo original de 4.259.498.761 bytes			
Tempo	gzip	bzip2	Xz
Real	8m11,214s	27m47,225s	69m21,364s
User	0m00,000s	25m31,140s	62m10,812s
Sys	0m00,031s	00m07,077s	00m33,375s
Total (User + Sys)	0m00,031s	25m38,217s	62m44,187s

Figura 31: Tempos de compactação do arquivo mbox do orientador

## Arquivos mbox da Enron Corporation

Os testes de compressão também serão realizados utilizando arquivos *mbox* públicos divulgados pela *FERC Federal Energy Regulatory Commission*, a agência americana que regulamenta a energia. Esses emails pertenciam à empresa *Enron Corporation*, que foi investigada pela FERC em (2002) e divulgados com o objetivo de promover a transparência, e para que pudessem, também, serem usados em pesquisas acadêmicas em diversas áreas, inclusive na ciência da computação. Atualmente essa base de dados é a maior base de conteúdo corporativo e de domínio público no mundo. Portanto, será adotado o

"Enron Corpus", isto é, a coletânea de emails confiscados no escândalo da Enron.

A versão do "Corpus" a ser utilizada é disponibilizada aqui: <https://www.cs.cmu.edu/~./enron/> (página de histórico e introdução ao arquivo) e [https://www.cs.cmu.edu/~./enron/enron\\_mail\\_20150507.tgz](https://www.cs.cmu.edu/~./enron/enron_mail_20150507.tgz) (endereço do arquivo para download). Essa versão não está no formato mailbox e não contém os arquivos originalmente anexados às mensagens.

Após o download, é obtido um arquivo *tgz* (abreviação de "tar.gz") que deverá ser descompactado com *gunzip*, resultando em um arquivo que será renomeado para *enron.tar*.

O formato *tar* é gerado pelo programa de mesmo nome cuja função é aglutinar arquivos e diretórios separados em um único arquivo, para facilitar armazenamento e transmissão. Um compactador aplicado a um arquivo *tar* poderá se beneficiar de repetições e semelhanças entre os arquivos armazenados, porque essas repetições agora se dão dentro de uma mesma cadeia de caracteres.

O programa *tar* não é objeto de estudo neste texto, então será informado, apenas, que o arquivo acima pode ser aberto com o comando *tar xf enron.tar* (ou *tar xzf enron\_mail\_20150507.tgz*, se for usada a opção do *tar* para invocar por si mesmo o compactador de interesse).

Note que *enron.tar* tem o tamanho de 1.823.016.960 bytes (1.823.019.008 bytes em disco) enquanto o diretório resultante *maildir* tem 1.421.183.736 bytes (2.728.198.144 bytes em disco), 517.401 arquivos e 3.499 pastas. A discrepância entre os diversos números é de interesse para nosso

estudo e é devida à representação dos arquivos no disco rígido (ou qualquer sistema de memória) do computador. Note que são muitos arquivos, a imensa maioria deles bastante pequenos. Isso ocorre pois um arquivo é armazenado em blocos de memória com um tamanho pré-definido. Portanto, um arquivo pequeno ocupa um único bloco de memória, mas todo esse bloco é utilizado para esse arquivo. Ao se criar o único arquivo no formato *tar*, todos esses pequenos arquivos são representados juntos, melhor utilizando os blocos de memória, mesmo com o uso adicional de espaço devido aos delimitadores que o programa *tar* deve inserir.

O arquivo *enron.tar* é perfeito para os testes de compactação. Com o mesmo sistema Windows mencionado acima, obtemos:

Tamanho do arquivo Enron original: 1.823.016.960 bytes		
Compressor	Tamanho após compactação (bytes)	Taxa de compactação (%)
gzip	443.254.797	24,3
bzip2	293.119.836	16,1
xz	183.495.868	10,1

Figura 32: Dados de compactação dos arquivos mbox da Enron Corporation

Apuração dos tempos de compactação do arquivo original Enron de 4.259.498.761 bytes			
Tempo	gzip	Bzip2	Xz
Real	2m12.301s	8m34.546s	26m46.390s
User	0m00.000s	7m13.656s	26m00.109s
Sys	0m00.047s	0m02.202s	00m12.890s
Total (User + Sys)	0m00.047s	7m15.858s	26m12.999s

Figura 33: Tempos de compactação dos arquivos mbox da Enron Corporation

## Arquivo Os Lusíadas

A tabela a seguir mostra o tamanho final dos arquivos e suas respectivas taxas de compactação após a compressão do arquivo *Os Lusíadas*:

Tamanho do arquivo Os Lusíadas original: 349.152 bytes		
Compressor	Tamanho após compactação (bytes)	Taxa de compactação (%)
gzip	141.452	40,5
bzip2	112.375	32,2
xz	122.408	35,0

*Figura 304: Dados de compactação do arquivo Os Lusíadas*

Abaixo a tabela evidencia cada tempo (*real*, *user* e *sys*) gasto com a operação de compactação do arquivo Os Lusíadas:

Apuração dos tempos de compactação do arquivo original de 349.152 bytes			
Tempo	gzip	bzip2	xz
Real	0m00,053s	0m00,002s	0m00,003s
User	0m00,053s	0m00,000s	0m00,003s
Sys	0m00,000s	0m00,002s	0m00,000s
Total (User + Sys)	0m00,053s	0m00,002s	0m00,003s

*Figura 315: Tempos de compactação do arquivo os Lusíadas*



## Referências

ALLEN, J.D. et al. *The Unicode Standard: Version 6.1*. Edição do Autor, 2012.

ANDRADE, A.V. et al. *Linux: Comandos básicos e avançados*. Edição do Autor, 2015.

CAMÕES, L. Os *lusíadas*. Disponível em <https://www.gutenberg.org/ebooks/3333.txt.utf-8> (Projeto Gutenberg). Acessado em 27/12/2016.

COLLIN, L. *LZMA Utils*. Disponível em <http://tukaani.org/lzma/>. Acessado em 27/12/2016.

COLLIN, L. *XZ Utils*. Disponível em <http://tukaani.org/xz/>. Acessado em 27/12/2016.

GAILLY, J.; ADLER, M. *Gzip: Introduction*, 2003. Disponível em <http://www.gzip.org/#intro>. Acessado em 27/11/2016.

GUTENBERG PROJECT. *Free ebooks by Project Gutenberg*. Disponível em <https://www.gutenberg.org/>. Acessado em 27/12/2016.

QMAIL. *Mbox*, 2011. Disponível em <http://qmail.org/man/man5/mbox.html>. Acessado em 09/12/2016.

SEWARD, J. *Bzip2 and libbzip2, version 1.0.5: A program and library for data compression*. Disponível em <http://www.bzip.org/1.0.5/bzip2-manual-1.0.5.pdf>. Acessado em 27/11/2016.



THE INTERNET ENGINEERING TASK FORCE. *(MIME) Part One: Format of Internet Message Bodies*, 1996. Disponível em <<https://tools.ietf.org/html/rfc2045#section-6.7>>. Acessado em 09/12/2016.

THE INTERNET ENGINEERING TASK FORCE. *The application/mbox Media Type*, 2005. Disponível em <<https://tools.ietf.org/html/rfc4155>>. Acessado em 09/12/2016.

THE INTERNET ENGINEERING TASK FORCE. *The Base16, Base32, and Base64 Data Encodings*, 2006. Disponível em <<https://tools.ietf.org/html/rfc4648#page-4>>. Acessado em 09/12/2016.

THE LINUX INFORMATION PROJECT. *ASCII: a brief introduction*, 2006. Disponível em <<http://www.linfo.org/ascii.html>>. Acessado em 09/12/2016.

THE LINUX INFORMATION PROJECT. *Plain Text Definition*, 2007. Disponível em <[http://www.linfo.org/plain\\_text.html](http://www.linfo.org/plain_text.html)>. Acessado em 09/12/2016.

THE LINUX INFORMATION PROJECT. *SMTP definition*, 2005. Disponível em <<http://www.linfo.org/smtp.html>>. Acessado em 09/12/2016.

WIKIPEDIA. *MIME*. Disponível em: <<https://en.wikipedia.org/wiki/MIME>>. Acessado em 09/12/2016.

WIKIPEDIA. *Zip (file format)*. Disponível em: <[https://en.wikipedia.org/wiki/Zip\\_\(file\\_format\)](https://en.wikipedia.org/wiki/Zip_(file_format))>. Acessado em 27/11/2016.