



Linux

Introdução ao Sistema Operacional e Comandos Básicos do **Shell**

Introdução

- Linux é o nome dado tanto para o **Kernel** (núcleo de um sistema operacional) quanto para o **sistema operacional** que roda sobre ele.
- Desenvolvido por **Linus Torvalds** e inicialmente distribuído em **1991** como um pacote GPL simples contendo apenas o **Kernel** e *alguns utilitários básicos*. O próprio usuário deveria compilar e configurar outros programas, alguns deles essenciais, para que o sistema atendesse suas necessidades.
 - Sua fama de sistema operacional **apenas para técnicos** incitou a **distribuição** de pacotes mais completos, que poupassem alguns esforços na instalação do Linux e que podiam ser desenvolvidos e direcionados para **públicos e hardware específicos**.
 - Atualmente, dentre um grande número de **distribuições**, as **mais populares** são aquelas que contam com o conjunto de aplicativos e utilitários do **projeto GNU**, da **Free Software Foundation**, uma das maiores contribuidoras no desenvolvimento do **Linux** e da proliferação da filosofia do **software livre**.



Linus Torvalds

Por que Linux para Bioinformática ?

- **Linux** é **GRATUITO**, e a maioria das suas ferramentas também;
- **Maioria** das aplicações de **Bioinformática** é desenvolvida para Linux;
- **Alto desempenho** e **fácil controle de processos/uso** de recursos;
- Amplamente utilizado pela **comunidade científica**;
- Possibilidade de ser **modificado** significativamente para **interesses específicos**;
- Excelente suporte para **scripting** e **programação**;
- Excelente suporte para **clusterização, multiprocessamento, computação distribuída**;
- Número expressivo de ferramentas **Open Source/Free**;

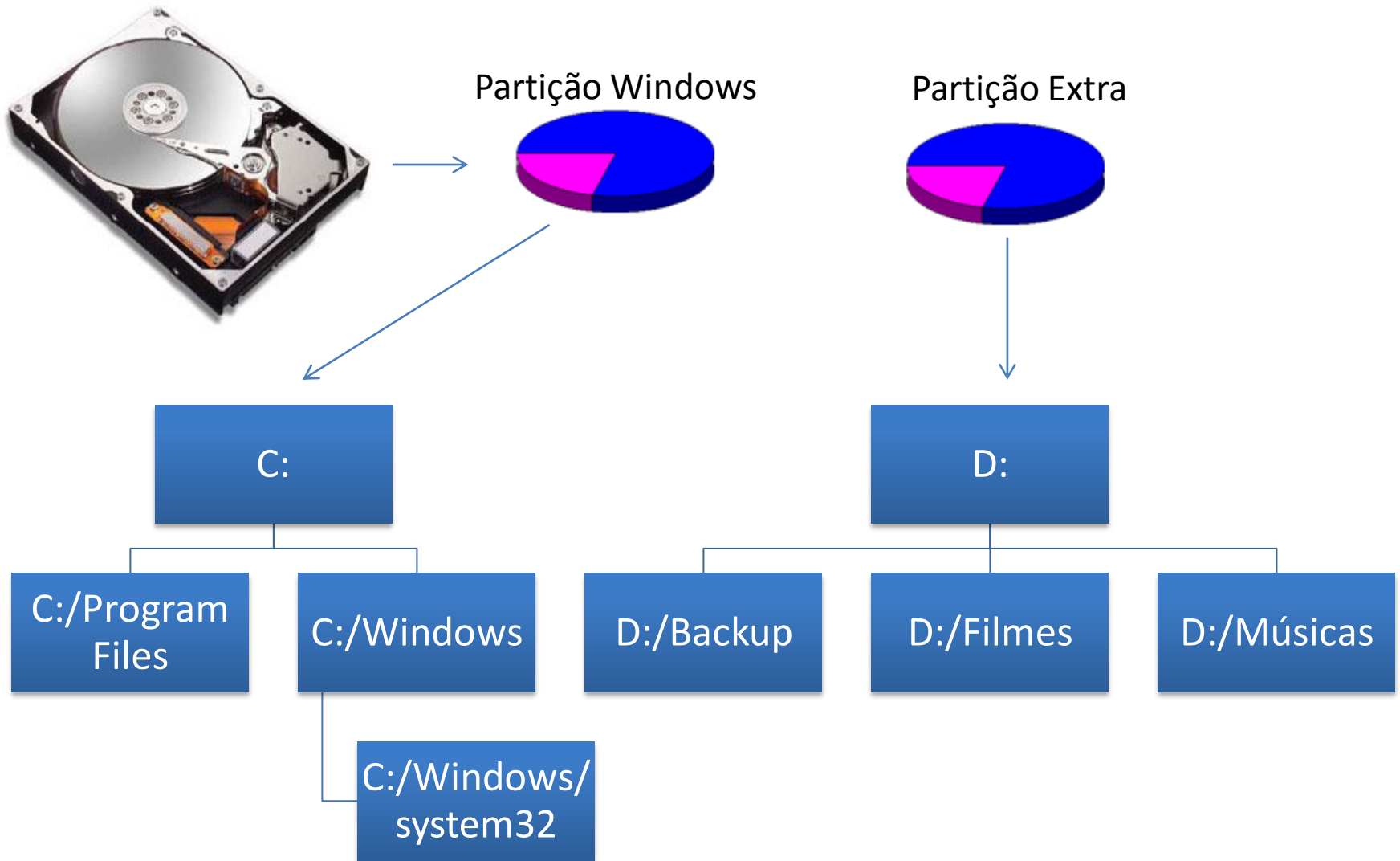


Distribuições Linux

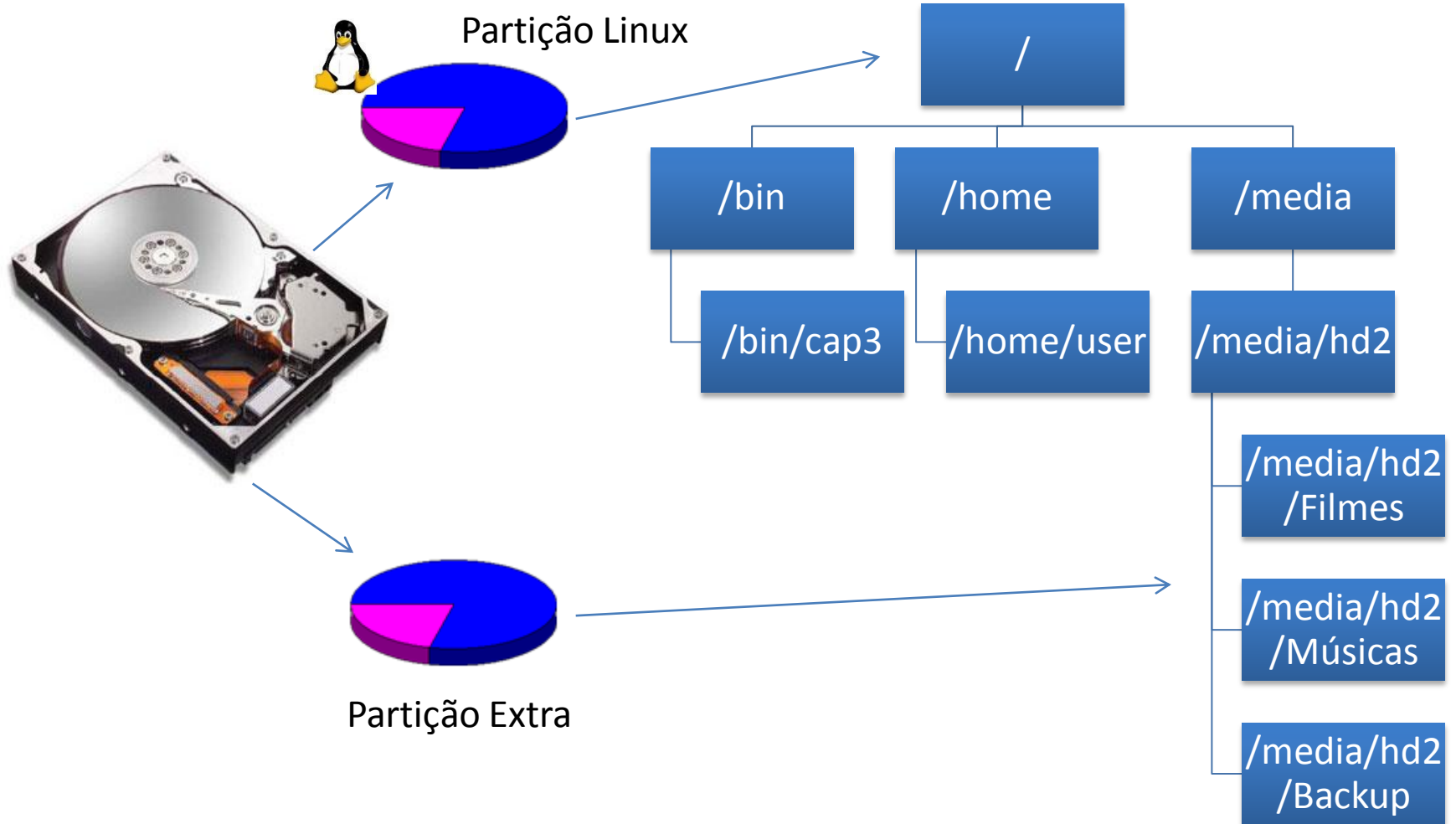
- Ubuntu
- Fedora
- Debian
- Slackware
- SUSE
- Gentoo
- Red Hat
- ...



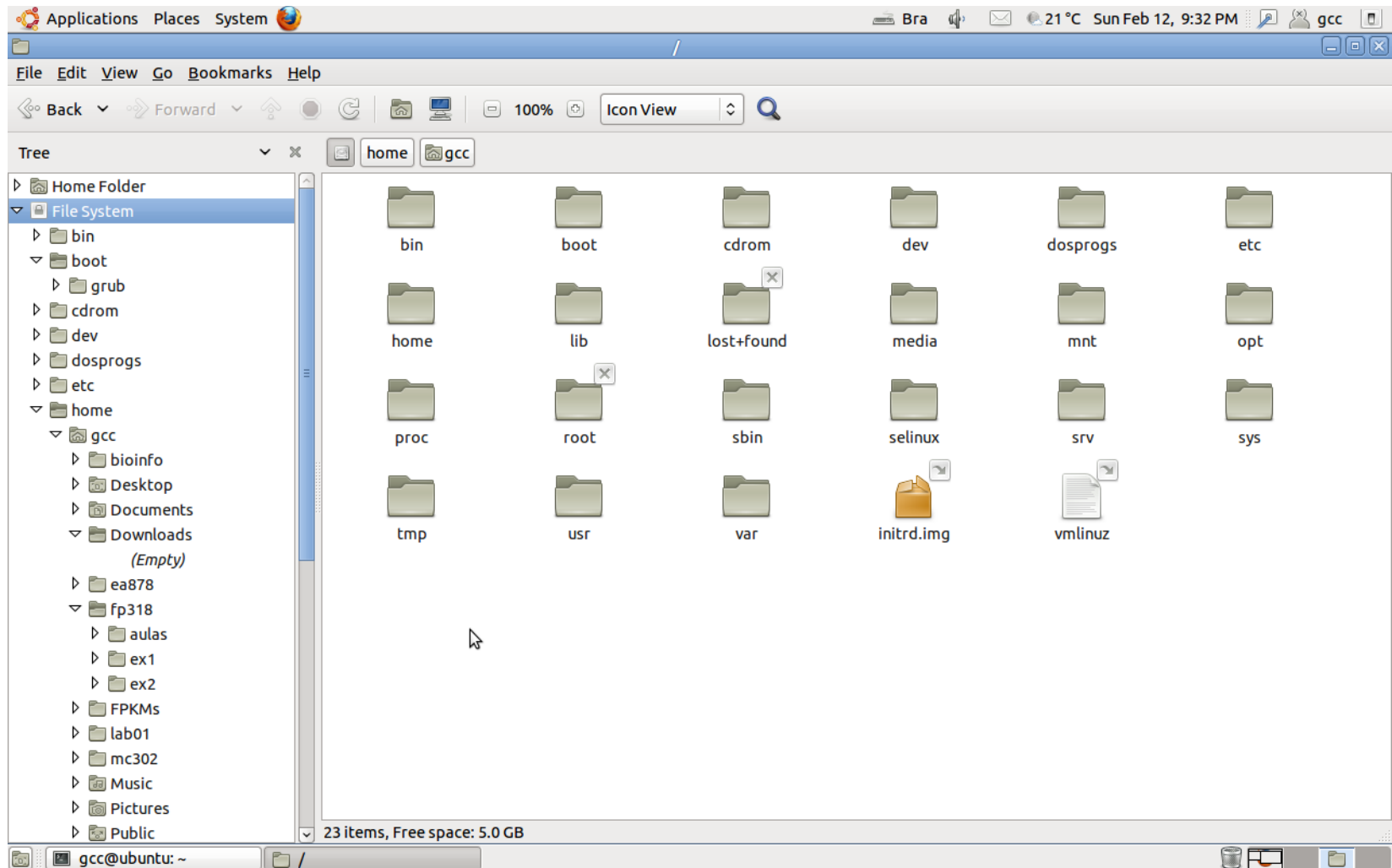
Sistema de Arquivos Windows



Sistema de Arquivos Linux

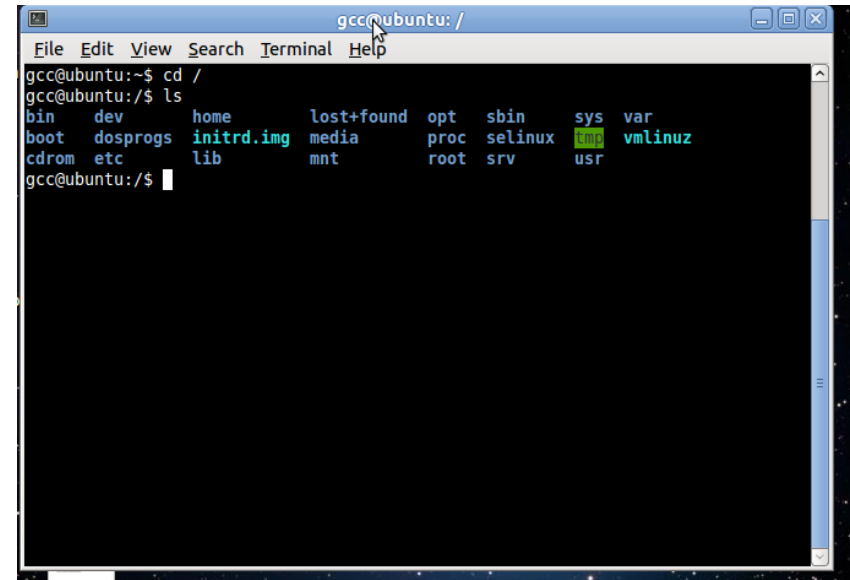


Gerenciador de Arquivos do X



Shell

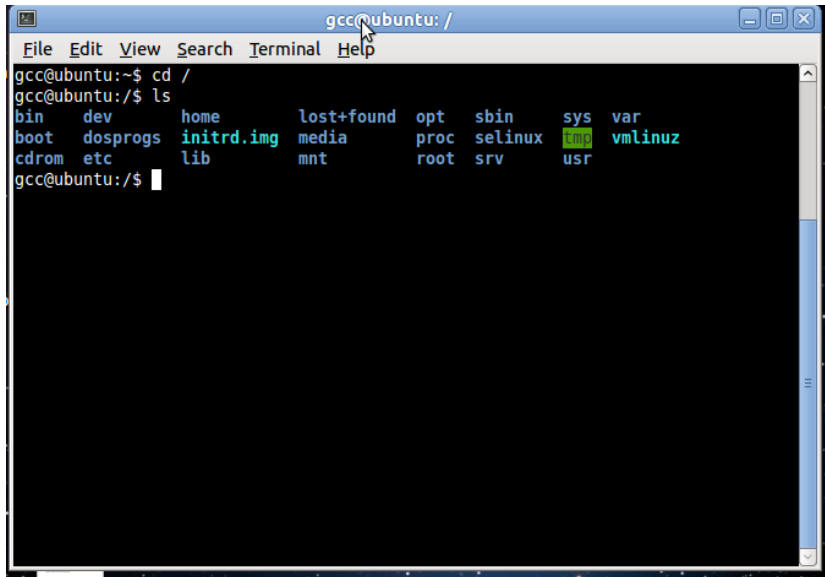
- O **X** pode proporcionar uma **experiência Linux equivalente** à da interface do gerenciador de janelas do Windows, com Mouse, movimentação e redimensionamento de janelas, ícones, botões e informações na tela.



- Muitas das **ferramentas avançadas de computação**, como as de bioinformática, **não são implementadas para um gerenciador de janelas** como o X, sendo necessária a utilização de um **interpretador de comandos Linux (Shell)** para que a ferramenta seja utilizada.

Shell

- **Ferramentas** tradicionais **ágeis, estáveis e confiáveis** como **cat, sed, grep, uniq, diff, awk, ... ***, **indispensáveis** para programadores Linux possuem **implementações** nativas com acesso por linha de comando (**Shell**).



```
gcc@ubuntu: /  
File Edit View Search Terminal Help  
gcc@ubuntu:~$ cd /  
gcc@ubuntu:/$ ls  
bin    dev    home    lost+found  opt    sbin    sys    var  
boot  dosprogs  initrd.img  media      proc   selinux  tmp    vmlinuz  
cdrom  etc     lib      mnt        root   srv      usr
```

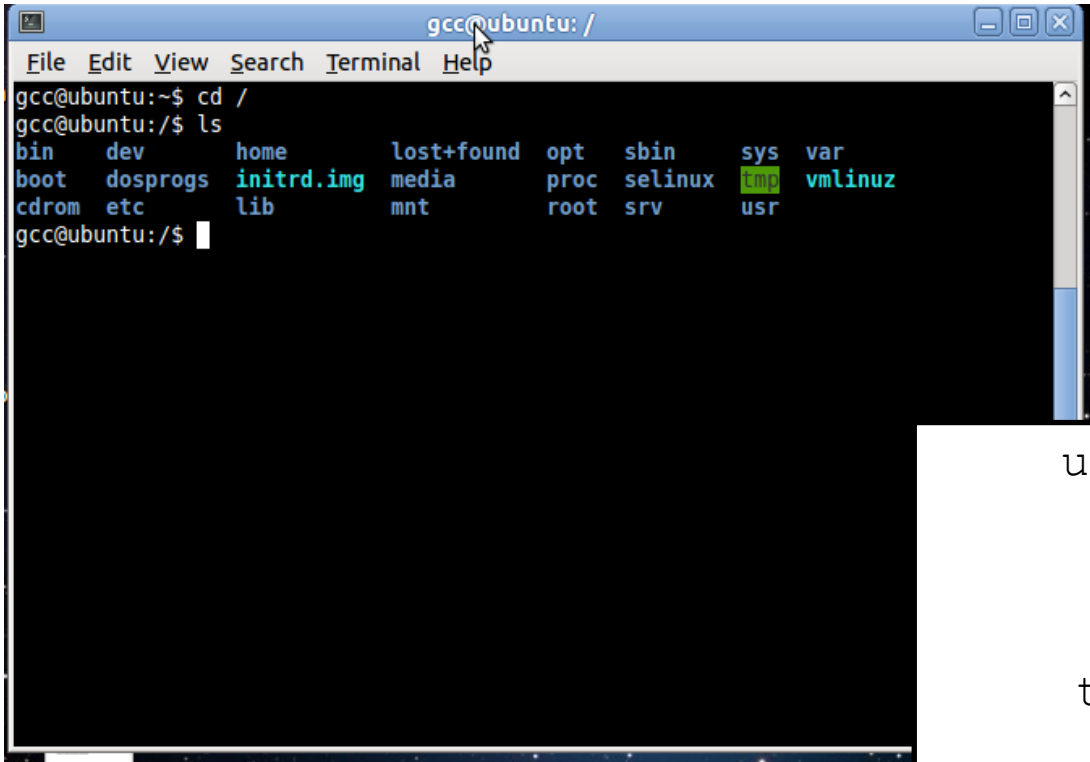
- O **Shell** torna-se uma ferramenta **poderosa** por possibilitar o **acesso** imediato a **qualquer arquivo** do sistema e a **integração** entre entradas e saídas de programas por **meta-caracteres ****.

**, ** Introduzidos nas próximas seções*

Shell

- Exemplos de Shell:
 - Bourne ou sh, C shell ou sch, Korn ou ksh e bash
- Abrir uma janela do Shell no X
 - Busque pelo aplicativo Terminal ou Gnome-terminal ou ainda Konsole no painel de aplicativos do seu gerenciador de janelas.
 - Atalho: ALT+F2
 - `"gnome-terminal"`
 - `"konsole"`
- Qual Shell estou usando?
`echo $SHELL`

Utilizando o Shell



```
gcc@ubuntu: /
File Edit View Search Terminal Help
gcc@ubuntu:~$ cd /
gcc@ubuntu:/$ ls
bin    dev    home   lost+found  opt    sbin    sys    var
boot   dosprogs  initrd.img media      proc   selinux tmp    vmlinuz
cdrom  etc      lib    mnt        root   srv     usr
```

usr@machine:dirname\$

Ex:

gcc@ubuntu:/\$

thiago@bioinfo03:~#

~ : Diretório Home

\$: Indica conexão sem
privilégios de root

: Indica conexão como root

Utilizando o Shell

- Padrão de Linha de Comando

```
usr@machine:path$ comando -[opcoes] [argumentos]
```

Ex1: gcc@ubuntu:~\$ rm -r projeto_antigo

Apaga projeto_antigo recursivamente. Caso seja um diretório, apaga também todos os seus arquivos e subdiretórios.

Ex2: \$ head -n 10 arquivo.txt

Imprime na saída padrão as primeiras 10 linhas de arquivo.txt

Por padrão os argumentos precedidos por - ou -- definem o tipo do próximo argumento ou uma opção.

No Ex1, -r é uma opção que faz com que o rm seja recursivo e projeto_antigo é o nome do diretório/arquivo.

No Ex2, -n precedendo o número 10 significa que head vai imprimir as 10 primeiras linhas de arquivo.txt

Utilizando o Shell - Navegação

- Comandos para navegação no sistema de arquivos

Cmd	Descrição	Argumentos Principais	
pwd	Mostra o caminho do dir. atual		
ls	Lista arquivos e pastas do diretório atual	-a ou --all -l ou --list -h	Lista arquivos ocultos Exibe detalhes Human readable
tree	Lista uma árvore de arquivos e pastas cuja raiz é a diretório atual	-d -L level	Lista apenas os diretórios omitindo Lista no máximo level níveis da árv.
cd	Muda de diretório		

- Atalhos especiais:
 - ~ : Diretório home do usuário
 - . : Diretório atual
 - .. : Diretório pai do diretório atual

Utilizando o Shell – Arquivos e Pastas

- Comandos para gerenciamento de arquivos e pastas

Cmd	Descrição	Exemplos	
mkdir	Cria um diretório	\$ mkdir dir1	Cria dir1
cp	Copia arquivo ou pasta	\$ cp ~/arq1 /bin	Copia arq1 para /bin
mv	Move arquivo ou pasta	\$ mv ~/arq1 /bin	Move arq1 para /bin
rm	Remove arquivo ou pasta	\$ rm ~/arq1	Remove arq1
touch	Modifica o time stamp de um arquivo, criando-o por padrão caso não exista	\$ touch a	Cria o arquivo a, ou modifica a timestamp do arquivo a para a data e hora atuais
ln	Cria um link para um arquivo	\$ ln -s /home/user/arquivo.txt link_name	Cria o link link_name no diretório atual que aponta para /home/usr/arquivo.txt

Utilizando o Shell – Exibição de Arquivos

- Como visualizar um arquivo pelo Shell ?

Cmd	Descrição
cat	Imprime o arquivo na saída padrão (stdout)
less	Pagina o arquivo na saída e exibe-o na saída padrão, oferecendo a possibilidade de navegar pelo arquivo.
more	Equivalente ao less, com menos recursos.
wc	Conta o número de linhas, caracteres e bytes de um arquivo

- Stdout ou Saída Padrão
Ex: Tela, Impressora, Arquivo
- Stdin ou Entrada Padrão
Ex: Teclado, Arquivo, Pipe
- Stderr ou Saída de Erro Padrão
Ex: Tela, Arquivo

Entrada e Saída de
Programas Linux

Utilizando o Shell – Editores de Texto

- Devido à limitação do terminal de exibir apenas caracteres, os editores de texto para o Shell devem ser do tipo WYSIWYM (what you see is what you mean). Formatação, inserção de imagens e gráficos devem ser realizadas por ferramentas externas que interpretam marcações no texto, como em arquivos HTML, Tex, ou LaTeX.

VI

Editor avançado de Texto presente em todas as distribuições Linux por estar padronizado pela Single Unix Specification. Concorrente do Emacs.

NANO

Editor de texto básico, com atalhos intuitivos, bem menos complexo que o VI.

Editores de Texto – Nano

- Editando um arquivo com o Nano

Abre o arquivo de nome filename, criando-o caso não exista

```
$ nano filename
```

Abre o arquivo o nano sem nenhum arquivo associado. O nome do arquivo deve ser indicado ao salvar o documento

```
$ nano
```

Os atalhos principais são exibidos na tela, o que torna o Nano muito fácil de usar.

\wedge G == Ctrl+G

Editores de Texto – Vi

- O Vi é um editor de texto que opera em dois modos principais:

Normal: Texto digitado é interpretado como comando.
É acessado, a partir de qualquer modo,
pressionando-se a tecla “esc” do teclado.

Inserção: Texto digitado torna-se parte do documento.
É acessado pressionando a tecla “i” do teclado no
modo normal

Editores de Texto – Vi

- Principais comandos do VI (modo Normal)

<http://pt.wikipedia.org/wiki/Vi>

Exercício – Prática 1

1. Entrar no diretório home do usuário
2. Dentro do seu diretório home, criar uma pasta com seu nome, em seguida entre nela
3. Criar as pasta teste1 teste2 e teste3 (no mesmo comando)
4. Criar as pastas subteste1 dentro de teste1
5. Mover a pasta subteste1 para a pasta teste2
6. Remover a pasta subteste1
7. Crie um arquivo vazio na pasta criada no passo 2
7. Copiar o arquivo ../pratica_linux/sequencias1.fasta para a pasta criada no passo 2 com o nome sequencias_copia.fasta, em seguida visualize o conteúdo do arquivo sequencias_copia.fasta (usar o vi) e insira seu nome o numero do seu grupo no arquivo “cabecalho”.
8. Adicionar o conteúdo do arquivo sequencias_copia.fasta no arquivo sequencias2.fasta (no mesmo dir, usando o comando cat) e concatená-los ao “cabecalho”.



Utilizando Shell – Ferramentas

- Várias ferramentas Unix simples e eficazes podem ser utilizadas para a manipulação de dados diretamente pelo Shell.

Vantagens:

- Relativa facilidade de uso
- Excelente documentação
- Versões estáveis há décadas
- Possibilidade de comunicação entre as ferramentas através do pipe
- Implementação extremamente otimizada

Ferramentas – Manual

- A documentação das ferramentas pode ser acessada usando o comando **man**.

Ex: `$ man ls`

Essa documentação é apenas para referência, não funcionando como um tutorial

- Mas se eu não souber qual comando procurar?

A ferramenta **apropos** pode ser utilizada para buscar nos manuais disponíveis por palavras-chave específicas.

Ex: `$ apropos -a create directory`

Ferramentas – Meta-Caracteres

- Alguns caracteres são interpretados como comandos no Shell. Esses caracteres são denominados meta-caracteres.

Meta-caracteres mais comuns:

*** : Qualquer string**

? : Qualquer caractere

[a-b] : Range de caracteres ASCII (letras ou números)

[abc, cde] : Lista de strings

Ferramentas – Meta-Caracteres

- Exemplos:

```
$ cp /bin/* .
```

Copia todos os arquivos da pasta /bin para o diretório atual

```
$ ls *.???
```

Copia todos os arquivos que possuem uma extensão de 3 caracteres. Ex: foo.txt, bar.foo

Ferramentas – Redirecionamento

- Alguns caracteres especiais, se usados na linha de comando, podem modificar o Stdin e o Stdout dos programas.

< : Modifica a entrada padrão do programa para um arquivo
Ex: `$ programa.pl < lista_de_genes.fasta`

> : Redireciona a saída padrão de um programa para um arquivo
Ex: `$ programa.pl > relatorio`

>> : Redireciona a saída padrão de um programa para o final de um arquivo (concatena)
Ex: `$ programa.pl >> relatorios`

Ferramentas para texto - grep

- O grep realiza busca por expressões por linha em arquivos de texto.

Exemplos:

```
$ grep ">" genes.fasta
```

Exibe na saída padrão as linhas do arquivo genes.fasta que contém o caractere ">".

```
$grep -l "GAATTC" *.seq > has_EcoRI.txt
```

Redireciona para o arquivo has_EcoRI.txt uma lista dos arquivos ".seq" do diretório atual que contém "GAATTC".

Ferramentas para texto – head/tail

- Head: Exibe na saída padrão as primeiras linhas de um arquivo

Ex:

```
$ head arquivo.txt
```

Exibe na saída padrão as primeiras 10 linhas de arquivo.txt

```
$ head -n 21 arquivo.txt
```

Exibe na saída padrão as primeiras 21 linhas de arquivo.txt

head/tail

- Tail: Exibe na saída padrão as primeiras linhas de um arquivo

Ex:

```
$ tail -n 15 arquivo.txt
```

Exibe na saída padrão as últimas 15 linhas de arquivo.txt

```
$ tail -n +2 arquivo.txt
```

Exibe na saída padrão todas as linhas do arquivo a partir da segunda.

Pipes

- A saída de um programa pode ser redirecionada diretamente para a entrada de outro. Essa ferramenta torna possível a combinação de vários processos para a realização de tarefas mais complexas.

```
$ grep "Toolik Lake" shaver_etal.csv | grep  
"Aug" > toolik_aug.csv
```

Redireciona para o arquivo "toolik_aug.csv" as linhas do arquivo que contém as strings "Aug" e "Toolik Lake".

Sort

- O Sort ordena as linha de um arquivo de acordo com parâmetros fornecidos pelo usuário.

Exemplos:

```
$ sort nomes > nomes.sorted
```

Escreve em nomes.sorted o conteúdo de nomes ordenado (tabela ASCII).

```
$ tail -n +2 arquivo.txt | sort -k 2 -n
```

Retira o cabeçalho da tabela que está no arquivo.txt e ordena as linhas numericamente de acordo com a segunda coluna.

Uniq

- Por padrão, o Uniq remove ocorrências de linhas duplicadas de um arquivo de texto. Para que isso ocorra o arquivo deve estar ordenado.

Exemplos:

```
$ sort nomes | uniq nomes.uniq
```

Escreve em nomes.uniq o conteúdo removendo os duplicados.

Cut

- O Cut fatia o arquivo de entrada verticalmente. É ideal para seleção de colunas de tabelas.

Exemplos:

```
$ grep ">" asb.fa |cut -c 2- > contigs.ids
```

Escreve os nomes dos contigs do arquivo asb.fa em contigs.ids

```
$ cut -f 5,7,9 -d "," ctd.txt > ctd.579.txt
```

Seleciona as colunas 5, 7 e 9 do arquivo ctd.txt, escrevendo-as em ctd.579.txt

Exercício – Cut/Sort/Uniq

- Conte o número de ocorrências de cada aminoácido do arquivo “aminoacidos “ utilizando os comandos CUT, SORT e UNIQ redirecionando o resultado para um arquivo.

Dica: utilize o manual para verificar as opções de cada uma das ferramentas.

awk

- O awk é uma linguagem de programação complexa que oferece recursos para processamento de texto, mas seu uso mais comum é a de seleção de colunas para reformatar arquivos.

Exemplos:

```
$ awk -F ',' '{print $2 " " $1}' tabela.csv  
| head -n 3 | less
```

Inverte a posição das colunas 1 e 2 da tabela.csv e exibe na tela pelo less.

sed

- Sed é utilizado para efetuar transformações em texto, como substituições.

Exemplos:

```
$ sed s/>Contig/>HybridContig/ assembly.fa  
> assembly.renamed.fa
```

Substitui a primeira ocorrência da string “>Contig” em cada linha do arquivo assembly.fa por “>HybridContig”.

```
$ sed s/t/u/g dna.seq > rna.seq
```

Monitoramento de Processos

Cmd	Descrição
top	Exibe a lista de processos em execução com info em tempo real
ps	Imprime no stdout a lista de processos em execução
kill	Envia um signal para um processo. Default KILL
df	Exibe informações sobre o espaço dos discos montados na máquina
du	Exibe o tamanho das pastas e arquivos de um diretório
free	Exibe informações sobre a memória física e swap utilizada pelo sistema

Nohup e &

- Nohup desvincula o processo da linha de comando do terminal. & executa o processo em segundo plano.

Ex:

```
$ nohup perl programa.pl
```

Ex:

```
$ ./sleep 20 &
```

Programa sleep é executado em segundo plano.