Tutorial 1

Introdução ao Linux

BIZ0433 - INFERÊNCIA FILOGENÉTICA: FILOSOFIA, MÉTODO E APLICAÇÕES.

Conteúdo

Objetivo

Este tutorial foi, em grande parte, copiado de uma apostila que encontrei nos servidores do IME/USP cujo objetivo era introduzir os sistema Lunix para principiantes. Como não constava nenhuma informação sobre os autores do documento, não pude dar devido crédito ao que foi inserido aqui.

O Objetivo deste tutorial é fazer com que o aluno entenda os princípios básicos do sistema operacional Linux, principalmente no que concerne ao uso de terminais (*i.e.*, modo texto) e comandos de linha na execução de tarefas básicas tais como, movimentação dentro do sistema operacional, criação de diretórios e arquivos, entre outros. Esse tutorial é fundamental para aqueles que estão matriculados nesse curso, uma vez que toda disciplina será baseada neste sistema operacional. No entanto, o aluno não deve ter a expectativa de que todos os comandos apresentados aqui serão absorvidos no primeiro contato. A proficiência nesses comandos vem com o tempo, e é um aprendizado que requer constante aprimoramento, pois sempre se aprende algo de novo. Portanto, este tutorial deverá ser sempre usado como referência para as demais aulas.

1.1 O que é Linux?

O termo Linux é usado em vários contextos com significados diferentes. A rigor, Linux é um kernel. No entanto, em alguns contextos, Linux significa sistema operacional (não qualquer sistema operacional, mas um que use o kernel Linux).

Sistema Operacional: é um *software* que serve de interface entre o computador e o usuário, gerenciando recursos (como memória, processamento, entre outros).

Kernel: é o núcleo ou cerne do sistema operacional (é a parte deste que fica mais "próxima" do hardware).

Você pode agora estar se perguntando se deve chamar apenas o kernel de Linux. Como dito anteriormente, a rigor, Linux é o kernel. Contudo, a expressão "sistema operacional Linux" tornou-se muito difundida. Outra pergunta pode ter surgido neste ponto: qual o nome do sistema operacional então? Mais uma controvérsia aqui. Quando algum usuário instala "o Linux", ele está instalando o kernel e mais uma série de outros *softwares* (*i.e.*, aplicativos ou programas). Grande parte desses aplicativos pertence a um projeto chamado GNU. Logo, o sistema operacional formado pelo kernel mais utilitários e aplicativos, como defendem alguns, deveria ser chamado de GNU/Linux.

1.1.1 Um breve histórico - Como surgiram o GNU e o Linux:

No ano de 1984, Richard Stallman iniciou o Projeto GNU, que tinha por objetivo criar um sistema operacional que fosse totalmente livre. Esse sistema operacional deveria ser compatível com outro sistema operacional - o UNIX (daí o nome GNU - GNU is Not Unix). No ano seguinte, Stallman fundou a FSF (*Free Software Foundation*), com o propósito de eliminar restrições de uso, cópia e distribuição de *software*.
Por volta de 1991, o sistema GNU estava quase pronto, exceto pelo kernel. Stallman estava trabalhando no desenvolvimento de um kernel chamado Hurd. Ao mesmo tempo, o finlandês Linus Torvalds havia criado um kernel compatível com as aplicações do projeto GNU. A esse kernel foi dado o nome de Linux. Atualmente, Linux tornou-se um termo genérico para se referir a sistemas operacionais "Unix-like" baseados no kernel Linux. Tornou-se, também, o melhor exemplo de Software Livre e de código aberto.

1.1.2 Software Livre e Licença GPL:

Na Seção anterior, foi dito que Stallman pretendia criar um sistema operacional livre e que o GNU/Linux era um exemplo de Software Livre. A definição de Software Livre, dada pela FSF é:

i. Executar o *software* com qualquer propósito (liberdade nº 0).

ii. Estudar o funcionamento do *software* e adaptá-lo às suas necessidades (liberdade nº 1).

iii. Redistribuir (inclusive vender) cópias do *software* (liberdade nº 2).

iv. Melhorar o programa e tornar as modificações públicas para que a comunidade inteira se beneficie da melhoria (liberdade nº 3).

Ao contrário do que as pessoas pensam, Software Livre (do inglês *Free Software*) não é sinônimo de gratuito. O que ocorre é uma confusão envolvendo a palavra "*free*" em inglês, que significa tanto gratuito como livre. Mas o sentido que Stallman queria dar era de "livre". De qualquer forma, a maioria dos *softwares* livres é distribuída de forma gratuita. Grande parte dos projetos de *software* livre (incluindo o GNU/Linux) é distribuída sob a GPL (*General Public License* - Licença Pública Geral), que é a licença idealizada por Stallman e que se baseia nas quatro liberdades citadas anteriormente. Com a garantia destas liberdades, a GPL permite que os programas sejam distribuídos e reaproveitados, mantendo, porém, os direitos do autor de forma a não permitir que essa informação seja usada de uma maneira que limite as liberdades originais.

1.1.3 Distribuições:

Distribuições Linux (também chamadas Distribuições GNU/Linux ou simplesmente distros) consistem em "pacotes" de *software* baseados no kernel Linux que incluem determinados tipos de *software* para satisfazer as necessidades de um grupo específico de usuários, dando assim origem a versões domésticas, empresariais e para servidores.
Exemplos de Distribuições Linux: Ubuntu, Debian, Slackware, Fedora, Red Hat, Arch, Gentoo, Mandriva, openSUSE etc.
Qual é a melhor distribuição é uma questão de necessidade e gosto. Apresentamos a seguir uma breve descrição de algumas distros, para que você possa ter uma ideia de suas principais características.

i. Debian:
A distro Debian (ou Debian GNU/Linux) é desenvolvida pelo Projeto Debian, um grupo de voluntários mantido por doações através da organização sem fins lucrativos Software in the Public Interest (SPI).
Debian baseia-se fortemente no projeto GNU e tem como principais características um alto compromisso com estabilidade e segurança bem como uma grande facilidade no que concerne à instalação de programas, através de um gerenciador de pacotes completo (dpkg) e sua interface (apt), utilizados amplamente em outras distribuições.
A última versão estável desta distro é 7.8 de 10 de janeiro de 2015.

ii. Red Hat Entreprise Linux:
Red Hat Enterprise Linux é uma distro criada pela empresa norte-americana Red Hat. O foco desta distribuição é o mercado corporativo, incluindo versões para servidores e para desktops.
O Red Hat Enterprise Linux não possui um ciclo de lançamentos fixo: a versão atual é a 7 de

2015.

iii. Slackware:

Simplicidade e estabilidade são duas características marcantes na distribuição do Slackware. Muito comum em servidores, procura ser uma distribuição "leve", praticamente sem enfeites e rápida, muito apreciada por usuários mais experientes.
Encontra-se atualmente na versão Slackware 14.1.

iv. Ubuntu:

Ubuntu é uma distro GNU/Linux baseada na distro Debian e é patrocinada pela Canonical. A proposta do Ubuntu é oferecer um sistema operacional que qualquer pessoa possa utilizar sem dificuldades, independentemente de nacionalidade, nível de conhecimento ou limitações físicas (a palavra Ubuntu é de origem africana e significa "humanidade para os outros"). Essa distro oferece um ambiente atualizado e estável, focado na usabilidade e na facilidade de sua instalação.
A cada seis meses, uma nova versão da distro é lançada, a versão atual com suporte longo prazo é Ubuntu 14.04.2 LTS (Trusty Tahr). Os números 14 e 4 são, respectivamente, o ano e o mês do lançamento da versão e a sigla LTS significa "*long time support*" que é geralmente de 5 anos.

1.1.4 Aplicativos, diretórios e arquivos:

i. Aplicativos:

Basicamente, para qualquer programa que você utilizava no Windows, existe uma alternativa no GNU/Linux. Por outro lado, vários dos aplicativos inicialmente concebidos sob "GNU/Linux" também estão disponíveis na versão para Windows (*e.g.*, VLC). Caso você queira saber quais programas são equivalentes entre Linux e Windows, consulte esta página.

ii. Visão geral da organização dos arquivos no Linux:

Grosso modo, pode-se dizer que, no Linux, tudo é arquivo. Se há algo que não seja um arquivo, então este algo é um processo. No GNU/Linux (como no UNIX), não há diferença entre arquivo e diretório, uma vez que um diretório é apenas um arquivo contendo nomes de outros arquivos. Imagens, músicas, textos, programas, serviços e assim por diante são todos arquivos. Dispositivos de entrada e saída, e geralmente, todos os dispositivos, são considerados como arquivos. Todos estes arquivos estão organizados de acordo com uma hierarquia, isto é, há critérios que prevêm os principais diretórios e seu conteúdo (veja Figura 1.1). Estes critérios são definidos por um padrão, o FHS (*Filesystem Hierarchy Standard*).

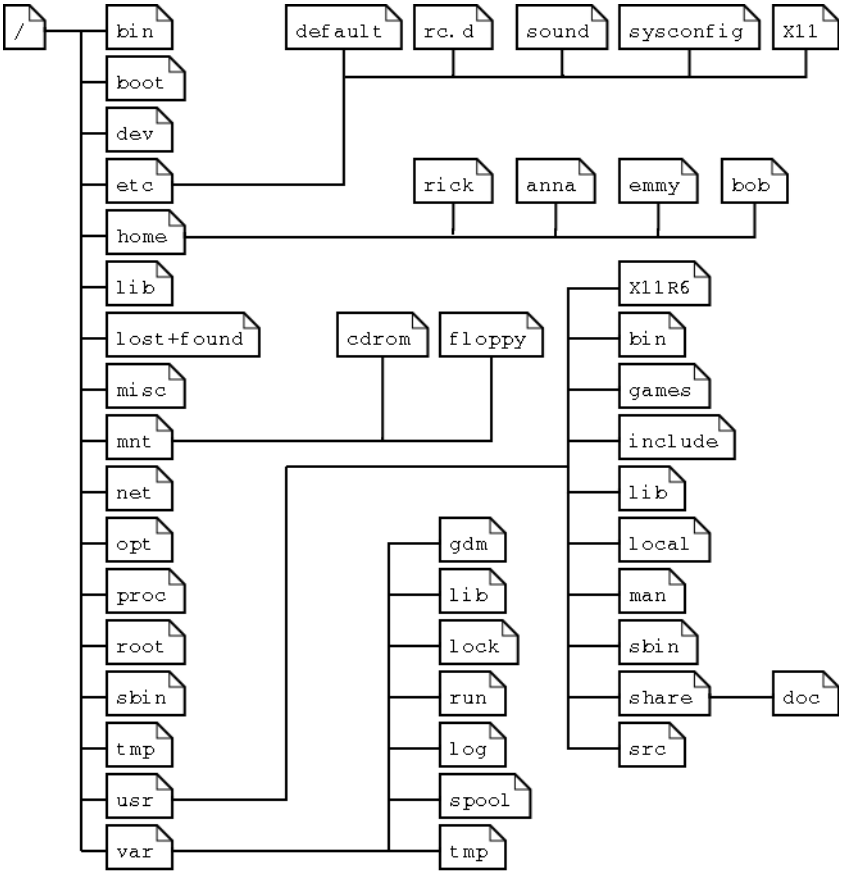

Figura 1.1: Estrutura hierárquica do *file system* do Linux.

No topo da hierarquia de arquivos fica o chamado diretório raiz (ou, mais apropriadamente, diretório *root*), pois a estrutura de diretórios é chamada também de "Árvore de Diretórios". Vejamos alguns exemplos citando os principais diretórios do sistema:

/: Chamado diretório root, é o diretório principal do sistema. Dentro dele estão todos os diretórios do sistema.

/bin: Contém comandos e programas essenciais para todos os usuários (alguns desses comandos serão tratados adiante).

/boot: Contém arquivos necessários para a inicialização do sistema.

/dev: Contém referências para todos os dispositivos, os quais são representados como arquivos com propriedades especiais.

/etc: Contém arquivos de configuração.

/home: Contém os diretórios dos usuários.

/lib: Contém bibliotecas (que são subprogramas ou códigos auxiliares utilizados por programas) essenciais para o funcionamento do Linux, e também os módulos do kernel.

/media: Contém subdiretórios que são usados como pontos de montagem para mídias removíveis, cdroms, discos externos e pen drives, entre outros.

/root: Este é o diretório "home" do super usuário (usuário root). Não confundir com o diretório root, o /. O diretório /root contém os arquivos do usuário root. O diretório / é o topo

da hierarquia de arquivos. **Usuário root** é o administrador do sistema, possui acesso a todos os comandos e arquivos.

/tmp: Contém arquivos temporários.

/usr: Contém programas, bibliotecas, entre outros arquivos.

/usr/bin: Contém os binários de programas não-essenciais (os essenciais ficam no /bin).

/usr/src: Contém os códigos-fonte de alguns programas instalados no sistema.

/var: Contém arquivos "variáveis", como logs, base de dados.

/var/log: Como o próprio nome diz, possui arquivos de log. **Arquivo de log:** é um arquivo que armazena registros de eventos relevantes de um programa ou do sistema.

/var/run: Contém informações sobre a execução do sistema desde a sua última inicialização.

iii. Caminho absoluto vs. Caminho relativo:

Caminho de um diretório (ou de um arquivo) é composto pelos diretórios que devemos percorrer até chegar a ele. Vamos diferenciar caminho absoluto de caminho relativo por meio de um exemplo.

Consideremos, por exemplo, o diretório xinit/. Consideremos que este diretório encontra-se dentro de um outro diretório, o diretório X11/. Este X11/, por sua vez, está dentro do diretório etc/, que, finalmente, está sob o diretório root, o /.

O raciocínio inverso seria: temos o / e dentro o etc/ (/etc/), e dentro o X11/ (/etc/X11) que contém o xinit/ (/etc/X11/xinit). Logo, "/etc/X11/xinit/" é o **caminho absoluto** ("*absolut path*") para o diretório xinit, ou seja, são os diretórios que devemos percorrer, começando pelo /, até o diretório xinit/.

Consideremos agora os mesmos diretórios do caso anterior (/etc/X11/xinit/). Suponhamos agora que estamos no diretório etc/. Para dizer qual é o caminho do diretório xinit/, bastaria dizer apenas X11/xinit - este é o **caminho relativo** ("*relative path*") do diretório (em relação ao diretório /etc – é seu **diretório de trabalho**). Se estivéssemos no diretório X11/, o **caminho relativo** seria simplesmente xinit/.

Em suma, caminho absoluto é aquele que utiliza toda a estrutura de diretórios, ao passo que o relativo toma um diretório como referência e define o caminho a partir daí.

iv. Permissões de acesso:

O Linux foi desenvolvido para ser um sistema multi-usuário. Isto significa que vários usuários podem ter configurações personalizadas, independentes das configurações dos demais usuários, bem como diferentes usuários podem executar tarefas ao mesmo tempo numa mesma máquina. Assim sendo, cada usuário pode querer negar ou permitir o acesso a determinado arquivo ou diretório. Por isso, existem as chamadas permissões de acesso do Linux: para impedir o acesso indevido de outros usuários ou mesmo de programas mal intencionados a arquivos e diretórios.

Mostraremos algumas destas permissões nesta seção. Mais adiante, mostraremos como manipulá-las.

v. Donos, grupos e outros:

No Linux, para cada arquivo são definidas permissões para três tipos de usuários: o **dono** do arquivo, um **grupo** de usuários e os demais usuários (**outros**, que não são nem o dono, nem pertencem ao grupo).

Dono: é o usuário que criou o mesmo, seu dono. Somente o dono e o usuário root podem mudar as permissões para um arquivo ou diretório.

Grupo: é um conjunto de usuários. Grupos foram criados para permitir que vários usuários tivessem acesso a um mesmo arquivo.

Outros: são os usuários que não se encaixam nos tipos de usuários supracitados.

vi. Tipos de permissões:

Os três tipos básicos de permissão para arquivos e diretórios são:

r (read): permissão de leitura para arquivos. Caso seja um diretório, permite listar seu conteúdo (com o comando ls, por exemplo - que será visto adiante).

w (write): permissão de escrita para arquivos. Caso seja um diretório, permite a gravação de arquivos ou outros diretórios dentro dele. Para que um arquivo/diretório possa ser apagado, é necessário o acesso à escrita (gravação).

x (execute): permite executar um arquivo. Caso seja um diretório, permite que seja acessado através do comando cd (você verá este comando também adiante, equivale a ”entrar” no diretório).

Em suma, para cada arquivo do sistema, são definidas permissões para o dono do arquivo, para um grupo de usuários e para os demais usuários. Essas permissões são de leitura, escrita e execução (r, w ou x). Você entenderá melhor estes conceitos adiante, mas tente familiarizar-se com eles desde já.

1.2 Modo texto

Não é apenas pelo modo gráfico que o usuário consegue interagir com o sistema. É possível fazer isso pelo modo texto, digitando comandos e nomes de programas para conseguir uma “resposta” do sistema. Por isso, o modo texto é também chamado de linha de comando.

É importante para um usuário do GNU/Linux aprender a trabalhar no modo texto por vários motivos: otimiza várias tarefas, existem alguns programas que só podem ser executados no modo texto e também porque o modo gráfico consome mais recursos computacionais.

Há duas formas de acessar o modo texto do sistema. Você pode acessar um terminal “puro”, pressionando as teclas “Ctrl+Alt+F1” simultanemamente. Na verdade, você pode substituir a tecla F1 por F2, F3, etc até F6. Isso lhe permite iniciar seis seções simultâneas no modo texto. Para retornar ao modo gráfico, basta pressionar as teclas “Ctrl+Alt+F7”! Veja como o modo texto se parece na Figura 1.2).

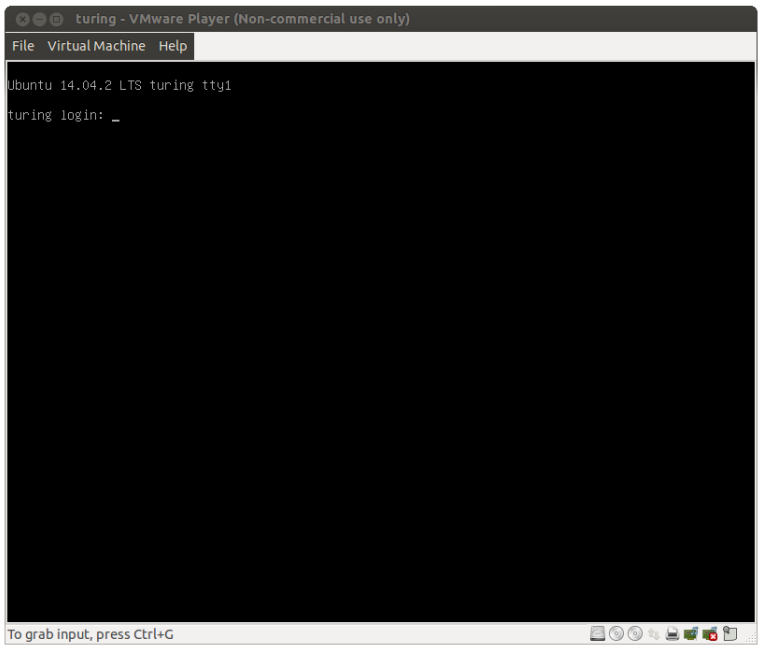

Figura 1.2: Login via terminal de texto (acesso com Ctrl+Alt+F1).

A segunda forma é usar um "emulador de terminal", isto é, dentro do modo gráfico, abre-se um programa que funciona com linha de comando. Na instalação disponibilizada no curso, o terminal está disponível na barra de ferramentas lateral (veja Figura 1.3). De forma geral, em Ubuntu 14.4.02 LTS, você pode acessar o terminal via Painel inicial (também na barra de ferramentas lateral) digitando o termo "terminal" (veja Figura 1.4).

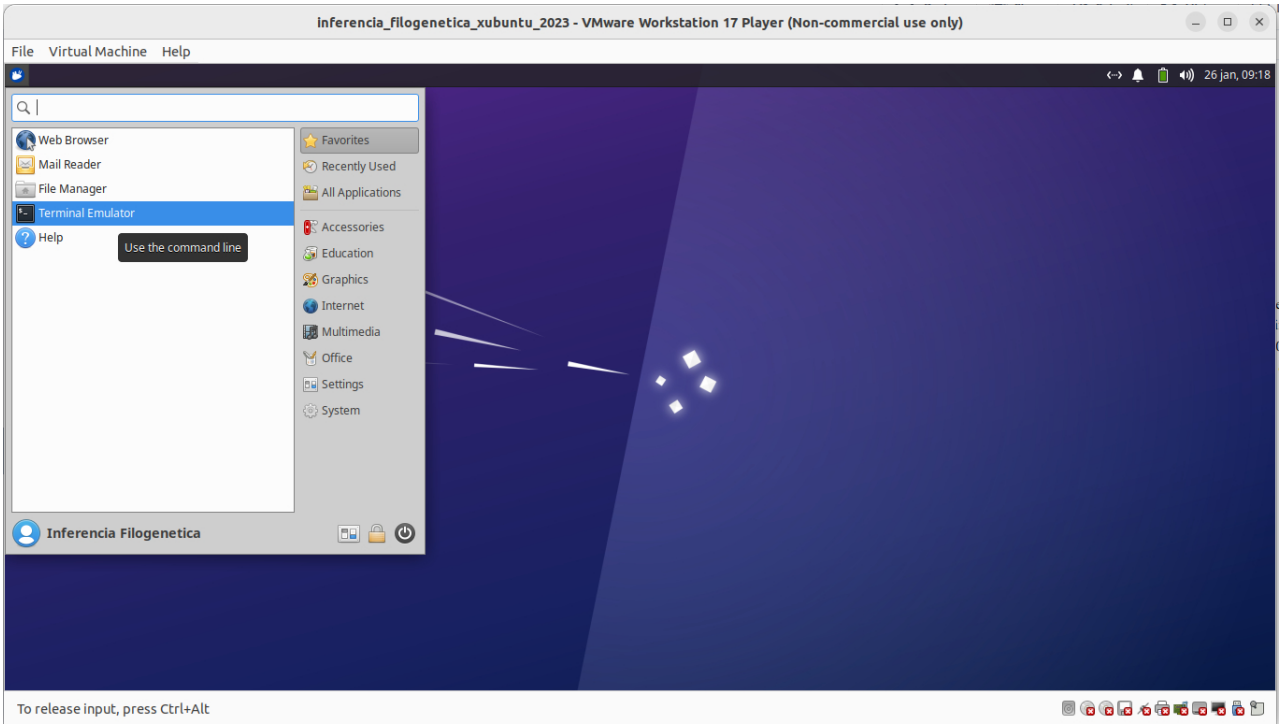

Figura 1.3: Acesso ao terminal via interface gráfica GNOME na barra de ferramentas lateral. [exemplo: imagem do Xubuntu, edição de 2023]

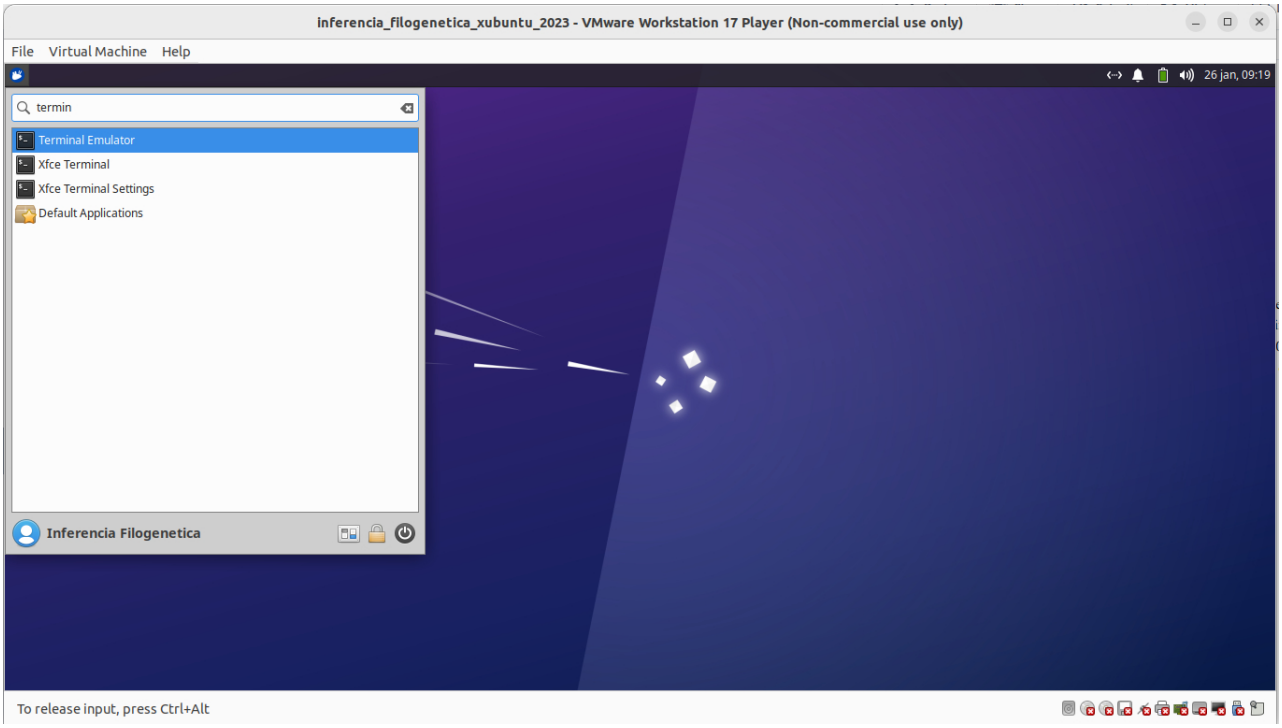

Figura 1.4: Acesso ao terminal via interface gráfica GNOME na barra de ferramentas lateral usando o Painel inicial. [exemplo: imagem do Xubuntu, edição de 2023

1.2.1 SHELL:

De qualquer uma das duas formas, o que você verá rodando (após logar-se ou acessar o Terminal) é um programa chamado shell, que é um interpretador de comandos.

1.2.2 BASH:

O BASH (Bourne Again Shell) é o shell desenvolvido para o projeto GNU, da *Free Software Foundation*, que se tornou padrão nas várias distribuições Linux (incluindo Ubuntu).

Figura 1.5: Estrutura de comandos *std_in* representa *standard input*, informações de entrada para o comando; *std_out* representa *standard output*, resultado do processamento feito pelo comando.

1.2.3 COMANDOS:

Nesta seção, examinaremos alguns comandos simples do BASH. É importante que você saiba que não é preciso decorar os comandos apresentados. Para aprendê-los de fato, você deve ir praticando com os exercícios propostos e conforme a sua necessidade.
Uma noção sobre comandos é importante. Comandos executam tarefas. Não existe comando que não

faça nada! Para executar tarefas, comandos necessitam de material para executar a tarefa para o qual ele foi concebido. Esse "material" é informação, dados. O comando pegará essa informação e retornará o resultado do processamento, ou seja, o resultado de seu trabalho. Veja a representação desses conceitos na Figura 1.5. Todo comando requer implícita ou explicitamente uma informação de entrada, *stardard input*, que será processada e exibida como resultado, *stardard input*, na tela ou redireciona a um arquivo (veja Seção **??** abaixo). Guarde esta estrutura, ela será fundamental para você entender o conceito de PIPES no final deste tutorial (Seção **??**)

1.2.3.1 Prompt:

O prompt do BASH tem a seguinte aparência:

```
alan@turing:~$
```

No caso de `alan@turing:~$` **alan** é o nome do usuário, **turing** é o nome da máquina, "~" é o diretório em que o usuário se encontra, neste caso, "~" representa o diretório **home** do usuário, e o "**$**" é o símbolo do tipo de usuário (nesse caso, um usuário normal). Se fosse o usuário root (administrador do sistema), o símbolo seria "**#**".

1.2.3.2 Sintaxe dos comandos:

É importante lembrar que a linha de comando é *case sensitive*, isto é, diferencia letras maiúsculas de minúsculas. Portanto, "echo" é diferente de "Echo", que são diferentes de "ECHO". Isso também vale para nomes de diretórios e arquivos. Os comandos são, em geral, em letras minúsculas. Muitos deles aceitam argumentos. Os argumentos que começam com um (ou dois) "-" são opções. Por exemplo:

```
$ comando -opção1 -opção2 -opção3 argumento
```

Quando os argumentos forem arquivos ou diretórios, tanto o caminho absoluto como o relativo poderão ser usados. Esta linha de comando assume a priori que se algum arquivo faz parte dos argumentos, ele está no seu diretório de trabalho.
Outro ponto importante é que você pode digitar os comandos e nomes de arquivos ou diretórios pela metade e depois pressionar "Tab". O shell "tentará completar" o que falta para você. Se houver mais de uma opção para completar o que foi digitado, as alternativas possíveis serão mostradas. Este é um recurso que facilita muito o uso da linha de comando. Vamos então aos comandos, você deverá abrir um terminal para que faça alguns dos exercícios.

1.2.3.3 pwd (print working directory):

Mostra o nome e o caminho do diretório atual, ou seja, diretório em que o usuário está, o diretório de trabalho.

```
$ pwd
   $ /home/alan
```

1.2.3.4 ls (list):

Lista os arquivos e subdiretórios de um ou mais diretórios.
Sintaxe básica:

```
$ ls [opções] [diretório1] [diretório2] ...
```

Exemplos:

i. O comando abaixo lista os diretórios e arquivos do /:

```
$ ls /
```

ii. O comando abaixo lista os diretórios e arquivos do /etc:

```
$ ls /etc
```

iii. Para listar o conteúdo do / e do /etc, de uma só vez, use:

```
$ ls / /etc
```

Exercício 1.1

Liste o conteúdo do diretório /tmp.
Para listar o conteúdo do diretório atual, basta digitar apenas "ls". Se o usuário estiver em seu diretório home e digitar ls, a saída será os arquivos e diretórios contidos no /home/alan.

Suponha ainda que o usuário encontra-se em seu diretório home. Existe, dentro do home do usuário, um diretório chamado "Documentos". Se quisermos listar o conteúdo deste, podemos usar o comando

```
$ ls /home/alan/Desktop
```

mas também podemos usar o caminho relativo (lembra-se?):

```
$ ls Desktop
```

Opções:

-a (ou all): Lista todos os arquivos e diretórios, incluindo os ocultos. No GNU/Linux, os arquivos e diretórios ocultos começam por ".". Quando usamos o comando ls como anteriormente (sem nenhuma opção), esses arquivos não são listados.
Exemplo:
O comando abaixo listará todos os arquivos e diretórios contidos no barra, incluindo os ocultos.

```
alan@turing:~$ ls -a
```

Exercício 1.2

Liste todo o conteúdo do seu diretório home, incluindo os itens ocultos. (Quando fizer isso, você notará que dois itens "estranhos" foram listados: o "." e o "..". Eles representam, respectivamente, o diretório atual e o diretório acima. Se você estiver em seu diretório home e usar o comando "ls ../", o conteúdo do /home será listado).

-R: Lista o conteúdo de um diretório e dos subdiretórios, recursivamente. Quando você utiliza o comando ls, os arquivos e diretórios contidos num determinado diretório são mostrados. Usando a opção -R, serão listados os arquivos contidos num determinado diretório, e para cada subdiretório também serão listados os arquivos e diretórios nele contidos. E para cada um desses diretórios, também será listado todo o seu conteúdo e assim sucessivamente. Se você usasse "ls -R /", o conteúdo de todos os diretórios seria mostrado (não execute esse comando, pois o *output* é longo, ele está aqui apenas para que você entenda o que faz esta opção).

-l: Usa o formato longo para listagem, o que significa que serão listados detalhes sobre cada arquivo e diretório mostrado. Vamos examinar que detalhes são estes. O comando

```
$ ls -l /
```

imprime: