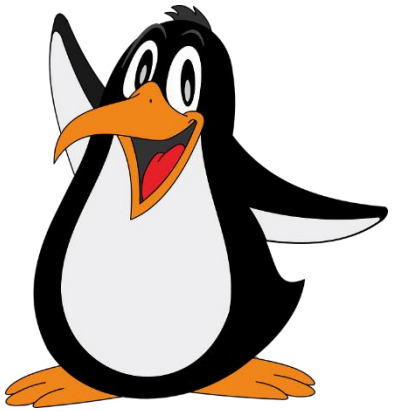
Sistemas Operacionais II

Linux: Introdução

Fabricio Breve
fabricio.breve@unesp.br
https://www.fabriciobreve.com

Agenda

- Histórico do UNIX e do Linux
- Desenvolvimento do Linux e Comunidade
- Distribuições do Linux
- Buscando informações sobre o Linux
- Programas Utilitários
- Interpretador de Comandos (Shell)
- Editores de Texto
- Compiladores (GCC)
- Programa da Disciplina
- Método de Avaliação



Histórico do UNIX

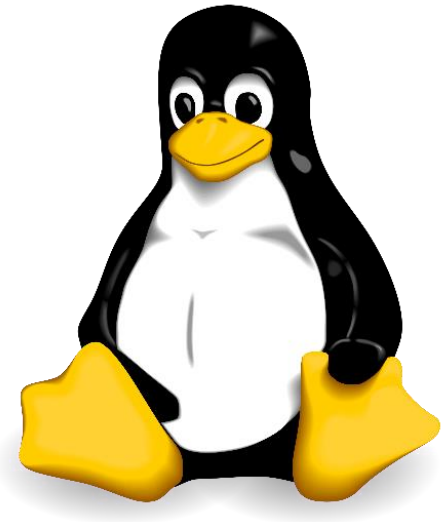
- 1969: origem como um projeto de pesquisa do AT&T Bell Labs
- 1976: Unix disponível gratuitamente nas universidades
- 1977: Universidade de Berkeley licencia o código da AT&T e faz suas próprias versões do Unix: BSD

Histórico do UNIX

- Com aceitação comercial, preço das licenças da AT&T sobem rapidamente
- Berkeley decide eliminar código da AT&T de seu Unix, mas perde as verbas para pesquisa de SO antes de concluir o projeto, deixando apenas o BSD-Lite.
- BSD-Lite dá origem aos vários UNIX BSD: BSD/OS, FreeBSD, NetBSD e OpenBSD
- Demais UNIX (HP-UX, Solaris, IRIX) são descendentes da linhagem AT&T original.

Histórico do Linux

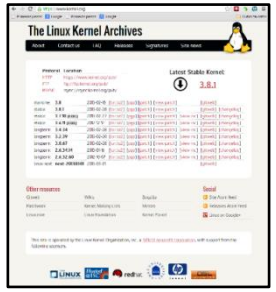
- Criado em 1991 por Linus Torvalds, aluno da Universidade de Helsinki, Finlândia.
  - O nome Linux provém de “Linus” e “UNIX”.
- O Minix serviu como inspiração para o projeto.
- Torvalds buscou conselhos na comunidade.
- Os desenvolvedores continuaram a apoiar o conceito de um novo sistema operacional gratuito.



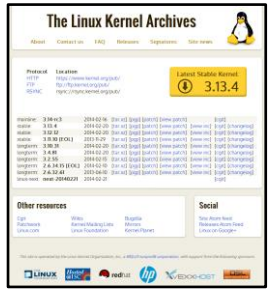
Histórico do Linux

- Kernel 1.0 lançado em 1994
- Versão atual (em 29/02/2024): 6.7.6

1S/2012 – 3.2.8



1S/2013 – 3.8.1

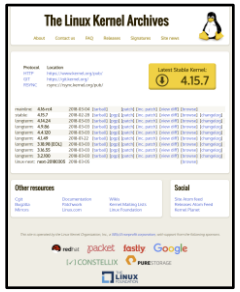


1S/2014 – 3.13.4

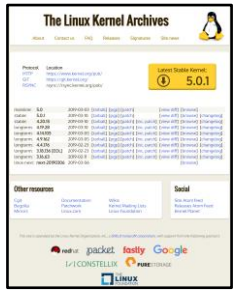
1S/2015 – 4.0

1S/2016 – 4.4.3

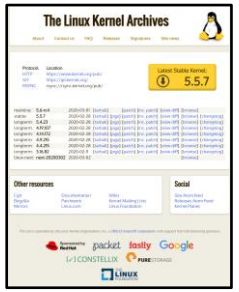
1S/2017 – 4.10.1



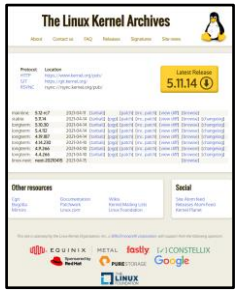
1S/2018 – 4.15.7



1S/2019 – 5.0.1



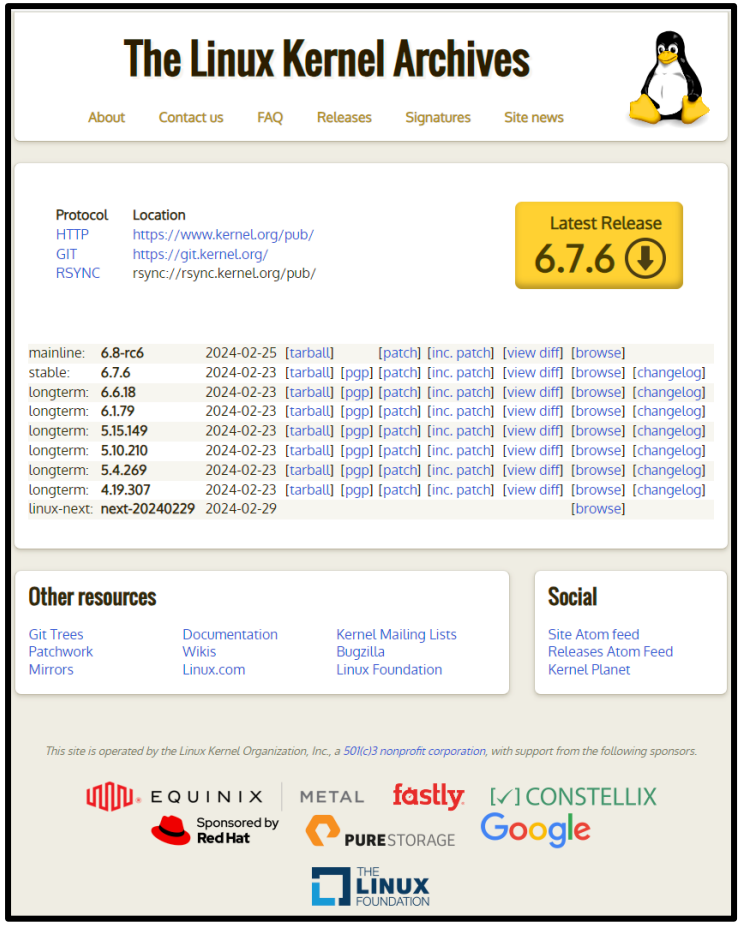
1S/2020 – 5.5.7



1S/2021 – 5.11.14

1S/2022 – 5.17

1S/2023 – 6.2.2



Distribuições Linux

- Permite que os usuários não familiarizados com os detalhes do Linux instalem e usem o Linux.
- Inclui softwares como:
  - Núcleo do Linux.
  - Aplicações de sistema (por exemplo, gerenciamento de conta de usuário, gerenciamento de rede e ferramentas de segurança).
  - Aplicações de usuário (por exemplo, GUIs, navegadores Web, editores de texto, aplicações de e-mail, bancos de dados e jogos).
  - Ferramentas para simplificar o processo de instalação.

Visão geral do Linux

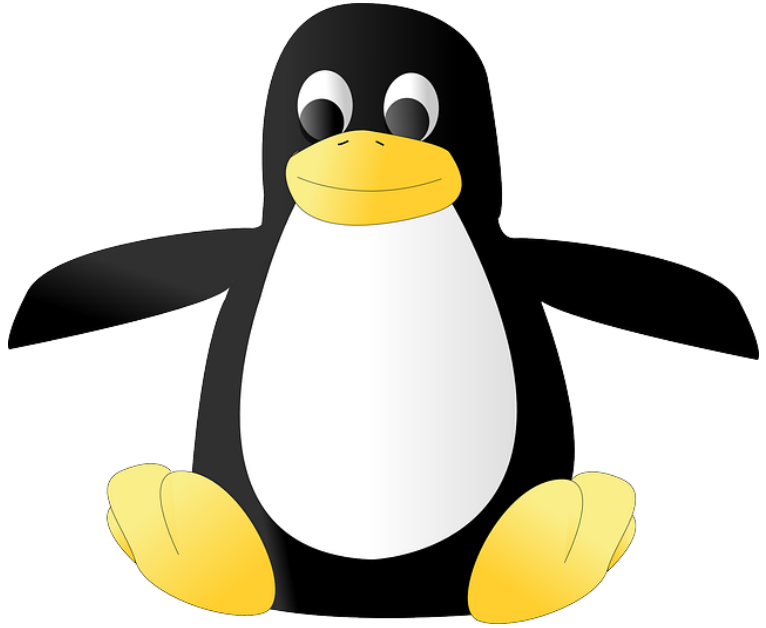
- Os sistemas Linux incluem, além do núcleo, interfaces com usuário e aplicações.
- Empresta do UNIX a abordagem em camadas do sistema.
- O sistema contém *threads* de núcleo para executar serviços.
  - Implementados como *daemons*, que permanecem adormecidos até serem acordados por um componente do núcleo.
- Sistema multiusuário
  - Restringe o acesso a operações importantes a um usuário com privilégios de superusuário (também chamado de raiz - *root*).

Desenvolvimento e comunidade

- Torvalds controla todas as modificações no núcleo.
- Ele conta com um grupo de mais ou menos 20 “tenentes” para gerenciar melhorias no núcleo.
- Quando o desenvolvimento do núcleo está para ser concluído:
  - Congelamento de característica: nenhuma característica nova é adicionada ao núcleo.
  - Congelamento de código: apenas os códigos que corrigem erros importantes são aceitos.
- Várias corporações apoiam o desenvolvimento do Linux.
- O Linux é distribuído sob a GNU Public License (GPL).
- O Linux é um software gratuito protegido por direitos autorais.

Algumas Distribuições Linux

Distribuição	Site Web
Debian	https://www.debian.org/
Fedora	https://fedoraproject.org/
Gentoo	https://www.gentoo.org/
Mageia	https://www.mageia.org/pt-br/
Manjaro	https://manjaro.org/
Mint	https://www.linuxmint.com/
MX Linux	https://mxlinux.org/
openSUSE	https://www.opensuse.org/
Red Hat Enterprise	https://www.redhat.com/en
Slackware	http://www.slackware.com/
SUSE Linux Enterprise	https://www.suse.com/
Ubuntu	https://ubuntu.com/



Red Hat

- **Red Hat Linux**: uma das primeiras distribuições Linux, primeira a implementar pacotes RPM, hoje descontinuada em favor da versão comercial.
- **Red Hat Enterprise Linux**: versão comercial, com assinatura anual (suporte pago)
- **Fedora**: substituiu a versão doméstica do Red Hat, projeto patrocinado pela Red Hat

SUSE Linux

- Bastante forte na Europa.
  - Tem origem na Alemanha.
- Comprado pela *Novell* em 2004
  - Passou a ter versão grátis disponível na web e versão comercial mais completa:
    - SUSE Linux Enterprise
    - openSUSE
- Novell e SUSE adquiridos pelo *The Attachmate Group* em 2011, que tornou SUSE um negócio independente.
- Attachmate comprado pela *Micro Focus International* em 2014
- SUSE vendido para a *Blitz 18-679 GmbH*, subsidiária da *EQT Partners* em 2019.

Debian

- Distribuição mais ligada ao projeto GNU
- Não é uma empresa, mas sim um projeto de mais de 1000 desenvolvedores voluntários em todo o mundo
- É base de outras distribuições: Knoppix e Ubuntu

debian

Ubuntu

- Lançado em 2004
- Derivado do Debian
- Focado em usabilidade
- Gratuito
- Patrocinado pela empresa *Canonical* do Reino Unido, cujo dono é o empresário sul-africano Mark Shuttleworth
  - Canonical obtém lucros vendendo suporte técnico

Mint

- Lançado em 2006
- Baseado no Ubuntu
  - Tem o objetivo de ser mais completo logo que instalado, incluindo softwares livres e de código aberto como *LibreOffice*, *Firefox*, *Thunderbird*, *HexChat*, *Pidgin*, *Transmission*, e *VLC media player*
  - Inicialmente também trazia softwares de terceiros, como *plugins* de navegadores, *codecs* de mídia, etc.
  - A partir da versão 18, ferramentas proprietárias não são mais incluídas, mas podem ser instaladas pelo usuário após o sistema estar em execução.

Slackware

- A mais antiga distribuição Linux que ainda é mantida
  - Lançada em 1993
- Já foi dominante, mas hoje é menos popular
- Simples, rápido e estável, ao custo de ser menos amigável (mais difícil de aprender)

Interface com o usuário

- Pode ser acessada por interpretadores de comando de console, como bash, csh e esh.
- A maioria das GUIs do Linux é composta de diversas camadas:
  - X Window System
    - Nível mais baixo.
    - Oferece mecanismos de camadas GUI mais altas para criar e manipular componentes gráficos.
  - Gerenciador de janela
    - Aproveita mecanismos da interface X Window System para controlar a busca, a aparência, o tamanho e outros atributos de janelas.
  - Ambiente de mesa (por exemplo, KDE, GNOME)
    - Oferece aplicações e serviços ao usuário.

Padrões

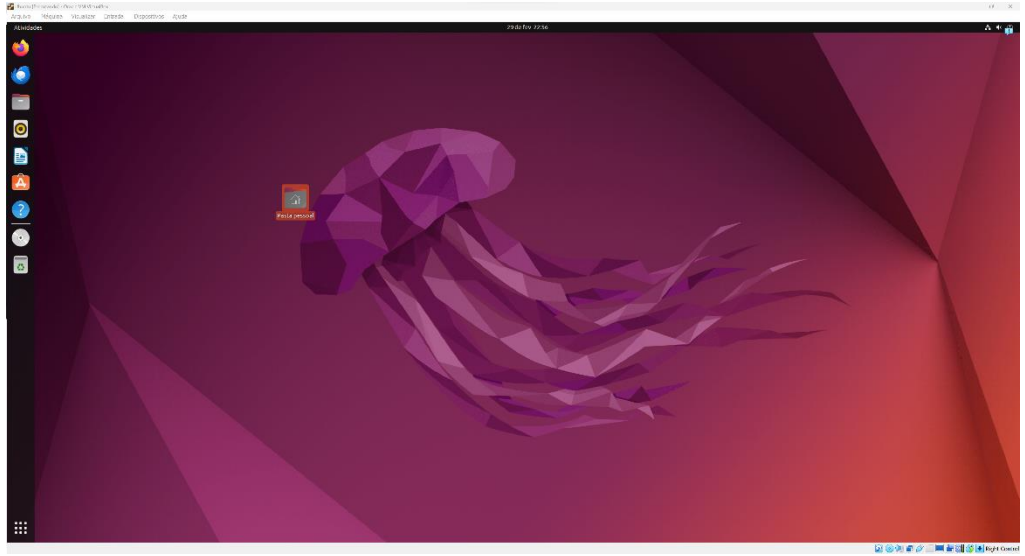
- Cada vez mais o Linux atende a padrões populares como o POSIX.
  - POSIX (Portable Operating System Interface): família de padrões definida pelo IEEE para manter compatibilidade entre sistemas operacionais
  - POSIX define uma interface de programação de aplicações (API), junto com *shells* de linha de comando e interfaces de utilitários, para compatibilidade de software entre variantes do Unix e outros sistemas operacionais.

Padrões

- Single UNIX Specification (SUS)
  - Conjunto de padrões que definem interfaces de programação de usuário e aplicação para sistemas operacionais UNIX, interpretadores de comandos e utilidades.
  - A versão 3 do SUS associa vários padrões (incluindo POSIX, os padrões ISO e versões anteriores do SUS).
- Linux Standards Base (LSB)
  - Projeto que busca padronizar o Linux de modo que as aplicações escritas para uma distribuição que siga o LSB compilem e comportem-se exatamente como em qualquer outra distribuição que também siga o LSB.

Linux em nossas aulas

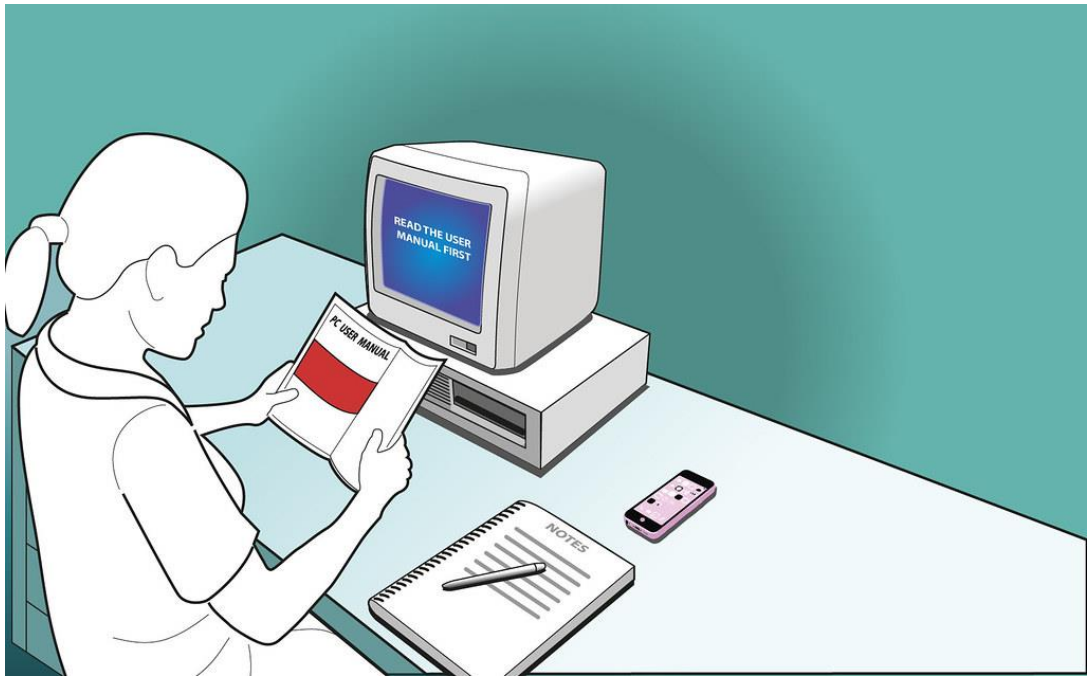
- Sem restrição de distribuição
  - Exemplos testados no **Ubuntu 22.04.2 LTS**
  - Interface gráfica não será necessária
    - Exceto se você quiser editar seus códigos em algum editor gráfico de sua preferência
- Alternativas para executar o Linux:
  - Nativamente
  - Máquina Virtual
    - Sugestão: Oracle VM VirtualBox
  - Subsistema do Windows para Linux



Guias de Instalação:
- Máquina Virtual: https://youtu.be/odNpkVzvwLw
- Subsistema do Windows para Linux: https://youtu.be/9V0JeFT6l3Y

Buscando informações: man e info

- Páginas de manual
  - `man` *`nome_do_comando`*
- Documentos Texinfo
  - `info` *`nome_do_comando`*
  - criado pois o comando *nroff* que formata páginas de manual é proprietário da AT&T (hoje já existe a versão GNU *groff*).
  - definem um segundo padrão desnecessário e complicado.



Páginas de Manual

- Distribuições do Linux incluem páginas de manual para maioria dos comandos, chamadas de sistema, e funções da biblioteca padrão
- As páginas são divididas em seções; para programadores, as mais importantes são:
  1. Comandos de Usuário
  2. Chamadas de Sistema
  3. Funções da Biblioteca Padrão
  4. Comandos de Sistema/Administrativos
- Exemplos:
  - **`man sleep`** → manual do comando *sleep*
  - **`man 3 sleep`** → manual da função *sleep* da biblioteca padrão

Vídeo demonstrando o uso do **man**:
https://youtu.be/QTw_FXRTf5k

Mais informações

- Arquivos de cabeçalho em **`/usr/include`**
  - Útil para verificar erros em chamadas de sistema.
    - Verificar se assinatura da função corresponde ao manual.
      - Eventualmente a página de manual pode estar desatualizada.
- Código-Fonte
  - O Linux tem código aberto e disponível livremente, podendo ser consultado.
    - O código-fonte pode não estar instalado no sistema por padrão.
    - O código fonte do núcleo normalmente está em **`/usr/src/linux`**

Localizar e instalar software: *which*

- Para saber se um aplicativo está no seu sistema use:
  - `which` *`nome_do_aplicativo`*
  - Exemplo:

```
which httpd
/usr/sbin/httpd
```

  - Busca no caminho de pesquisa (PATH)

```
fabricio@ubuntu-donald:~$ which gcc
/usr/bin/gcc
fabricio@ubuntu-donald:~$
```

Localizar e instalar software: *whereis* e *locate*

- `whereis httpd`
  - Intervalo de diretórios mais amplo.
- `locate httpd`
  - Pesquisa em um índice pré-compilado do sistema de arquivos
  - O banco de dados é construído através do comando *updatedb*
    - Normalmente executado diariamente pelo *cron*

```
fabricio@ubuntu-donald:~$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/share/man/man1/gcc.1.gz
fabricio@ubuntu-donald:~$ locate httpd
/etc/lighttpd
/etc/lighttpd/conf-available
/etc/lighttpd/conf-enabled
/etc/lighttpd/conf-available/90-javascript-alias.conf
/etc/lighttpd/conf-enabled/90-javascript-alias.conf
fabricio@ubuntu-donald:~$
```

Vídeo demonstrando o uso de *which*, *whereis* e *locate*: https://youtu.be/TyposdIs1-A

Localizar e instalar software (Red Hat, Fedora, SuSE, etc.)

- `rpm -q python`
  - Verifica se o pacote chamado python está instalado
  - Pacotes RPM são utilizados no Red Hat, Fedora, SuSe, etc...
  - Normalmente é mais fácil instalar o pacote binário compilado específico para sua distribuição
  - Você também pode buscar e compilar o código-fonte original
- `dnf install python`
  - Obtém e instala a versão mais recente do Python
  - DNF é um gerenciador de pacotes que usa o Sistema RPM
    - É um fork do YUM, resolvendo alguns problemas do mesmo

Localizar e instalar software: (Debian, Ubuntu, etc.)

- `apt install gcc`
  - Obtém e instala a versão mais recente do GCC.
  - Utilizado no Debian e seus derivados, como o Ubuntu e outros.
  - No Ubuntu e seus derivados o **snap** é uma outra opção de gerenciador de pacotes pré-instalado.

Executando programas como super usuário (root)

- `sudo`
  - Permite que usuários comuns obtenham privilégios de outro usuário, geralmente o root, para realizar tarefas específicas dentro do sistema de forma segura e sob controle do administrador.
  - O termo é uma abreviação de "substitute user do" (fazer substituição do usuário) ou "super user do" (fazer como superusuário).
  - Exemplo:
    - `sudo apt install gcc`
      - A instalação de pacotes tipicamente exige privilégios de super usuário.

Interfaces para o Linux

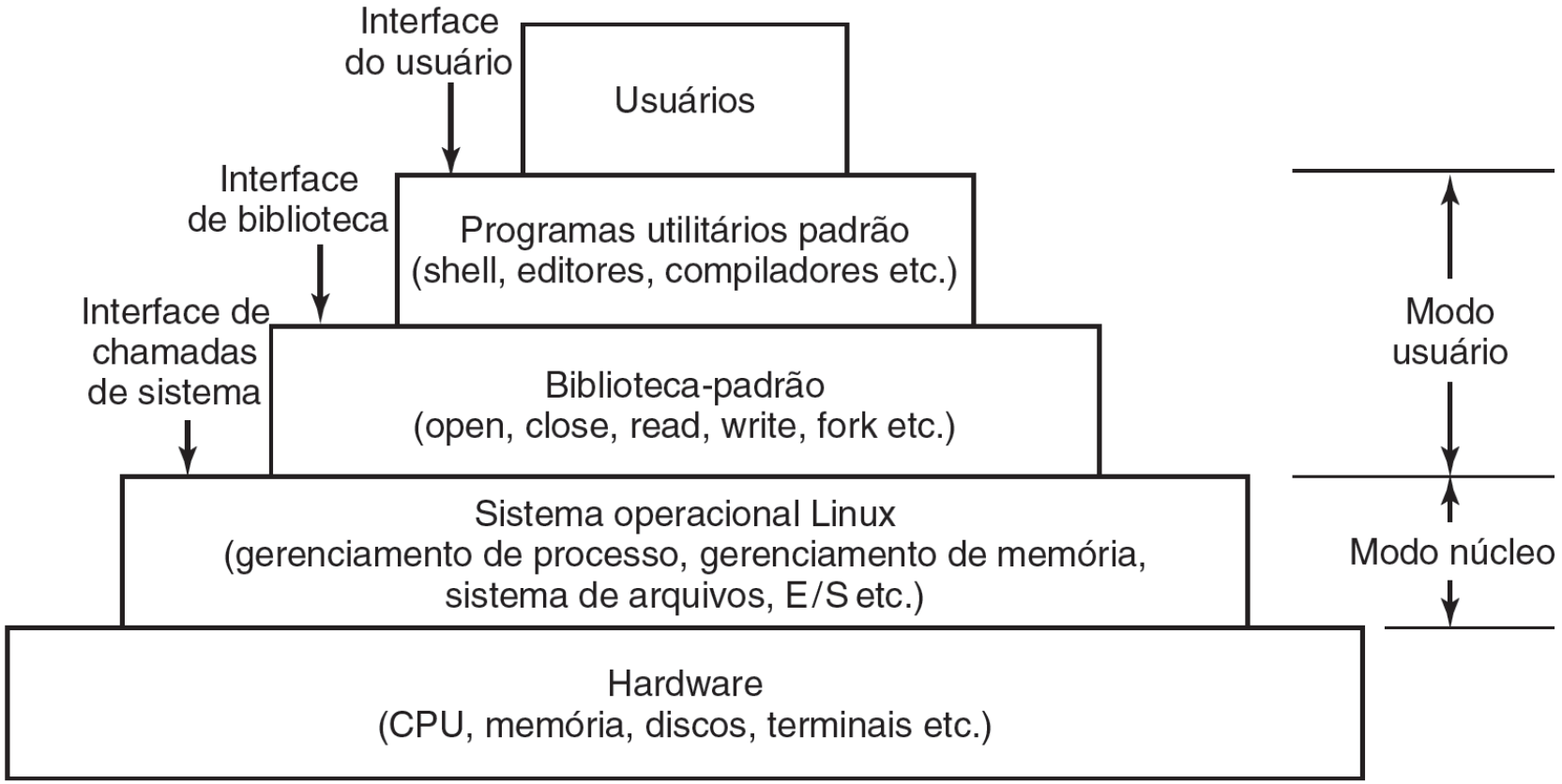


Figura 10.1 As camadas em um sistema Linux.

Programas utilitários do Linux

- Categorias dos programas utilitários:
  - Comandos de manipulação de arquivos e diretórios.
  - Filtros.
  - Ferramentas de desenvolvimento de programas, como editores e compiladores.
  - Processamento de texto.
  - Administração de sistema.
  - Miscelâneas.

Programa	Tipicamente
cat	Concatena vários arquivos para a saída-padrão
chmod	Altera o modo de proteção do arquivo
cp	Copia um ou mais arquivos
cut	Corta colunas de texto de um arquivo
grep	Procura um certo padrão dentro de um arquivo
head	Extrai as primeiras linhas de um arquivo
ls	Lista diretório
make	Compila arquivos e constrói um binário
mkdir	Cria um diretório
od	Gera uma imagem (dump) octal de um arquivo
paste	Cola colunas de texto dentro de um arquivo
pr	Formata um arquivo para impressão
ps	Lista os processos em execução
rm	Remove um ou mais arquivos
rmdir	Remove um diretório
sort	Ordena um arquivo de linhas alfabeticamente
tail	Extrai as últimas linhas de um arquivo
tr	Traduz entre conjuntos de caracteres

Tabela 10.1 Alguns dos programas utilitários Linux comuns, requeridos por POSIX.

Interpretador de Comandos (Shell)

- Bash é o mais comum e padrão na maioria dos sistemas
  - BASH é um acrônimo de Bourne Again Shell
    - Bourne é a shell original do UNIX
- Shell é um programa comum de usuário
  - Quando usuário digita uma linha de comando:
    - Extrai a primeira palavra, procura pelo programa correspondente e o executa
    - Suspende a si próprio até que o programa executado termine

```
fabricio@ubuntu-donald:~$ echo $SHELL
/bin/bash
fabricio@ubuntu-donald:~$ $SHELL --version
GNU bash, versão 5.0.17(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2019 Free Software Foundation, Inc.
Licença GPLv3+: GNU GPL versão 3 ou posterior <http://gnu.org/licenses/gpl.html>
.

Este é um software livre; você é livre para alterar e redistribuí-lo.
Há NENHUMA GARANTIA, na extensão permitida pela lei.
fabricio@ubuntu-donald:~$
```

Interpretador de Comandos (Shell)

- Comandos podem levar argumentos que são passados como cadeias de caracteres ao programa chamado.
  - Exemplos:
    - `cp src dest` (dois argumentos)
    - `head -20 file` (flag indicada com um traço)
    - `ls *.c` (wildcards)

Interpretador de Comandos (Shell)

- `sort` (Ctrl+D para fim de arquivo)
- `sort <in >out` (lê entrada de in e escreve em out)
- `sort <in >temp; head -30 <temp; rm temp`

  (lê entrada de in, grava saída em temp, imprime primeiras 30 linhas de temp e remove temp)
- `sort <in | head -30`

  (igual exemplo anterior, mas cria uma pipeline, dispensando arquivo temporário)
- `grep ter *.t | sort | head -20 | tail -5 > foo`

  [pega todas as linhas com 'ter' de arquivos *.t*, ordena-as, pega as 20 primeiras linhas da lista ordenada, pega as 5 últimas linhas (linhas 16-20 da lista anterior) e grava em *foo*

Vídeo demonstrando o uso do Interpretador de Comandos: https://youtu.be/Xxjxrrflthg

Interpretador de Comandos (Shell)

- `wc -l <a >b &`

  (wc – contador de palavras, flag `-l` indica que deve contar linhas, *a* é entrada, *b* é saída, tudo feito em segundo plano por causa do &)

- `sort <x | head -1 &`

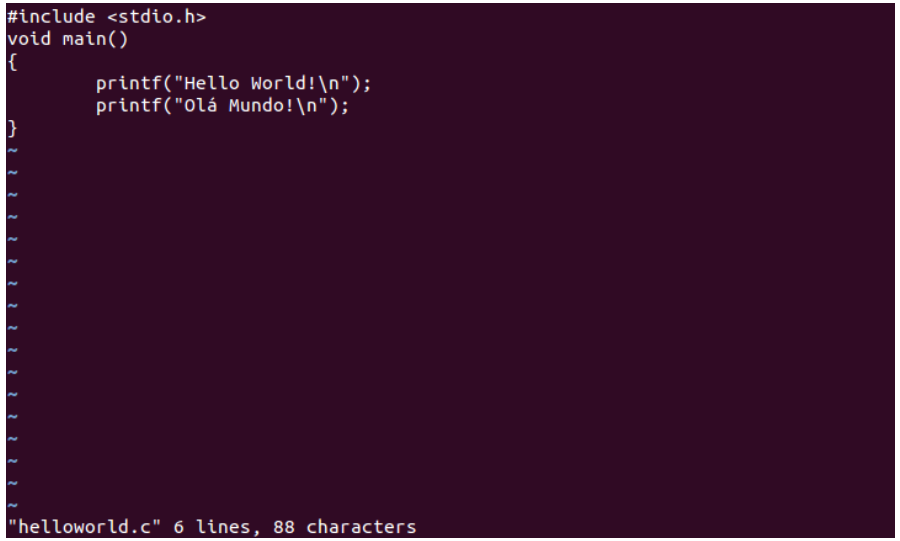
  (pipeline em segundo plano)

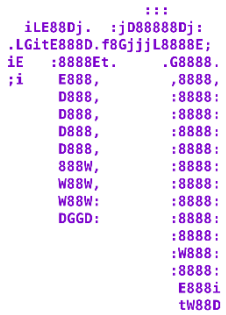
Interpretador de Comandos (Shell)

- É possível criar um arquivo texto com vários comandos de *shell* (*shell script*) e depois executá-lo chamando o *shell* com tal arquivo como entrada
  - Também é possível usar estruturas *if*, *for*, *while*, *case*, etc.
    - Berkeley C é um *shell* alternativo com sintaxe semelhante à linguagem C
    - Qualquer pessoa pode escrever seu próprio *shell* dando suporte a outras linguagens

Editores de Texto

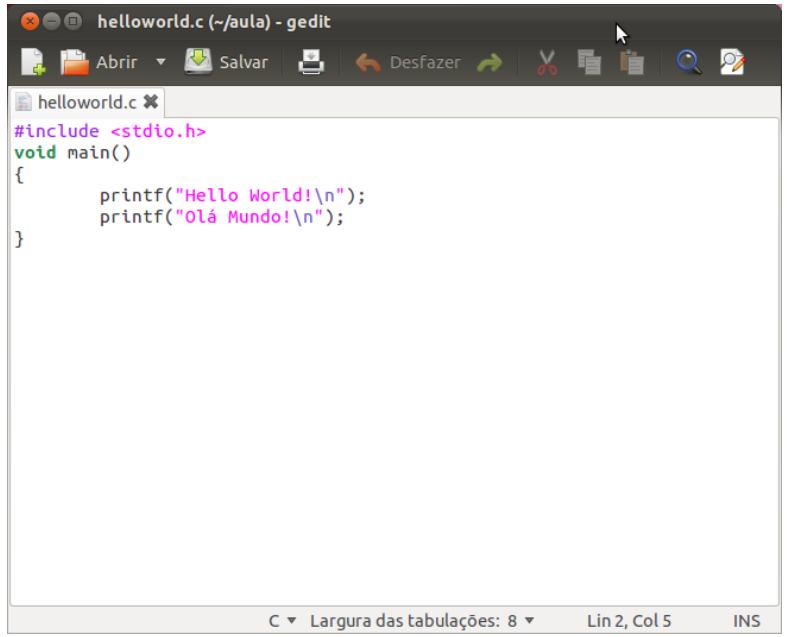
- Modo Texto
  - **vi**
    - Amplamente disponível
    - Difícil aprendizado
  - **vim**
    - Versão melhorada do vi
  - **pico**
    - Aprendizado mais fácil
    - Parte do **pine**
    - Destaque de sintaxe de C e outras linguagens
  - **nano**
    - Clone do pico, sem a sobrecarga do pine

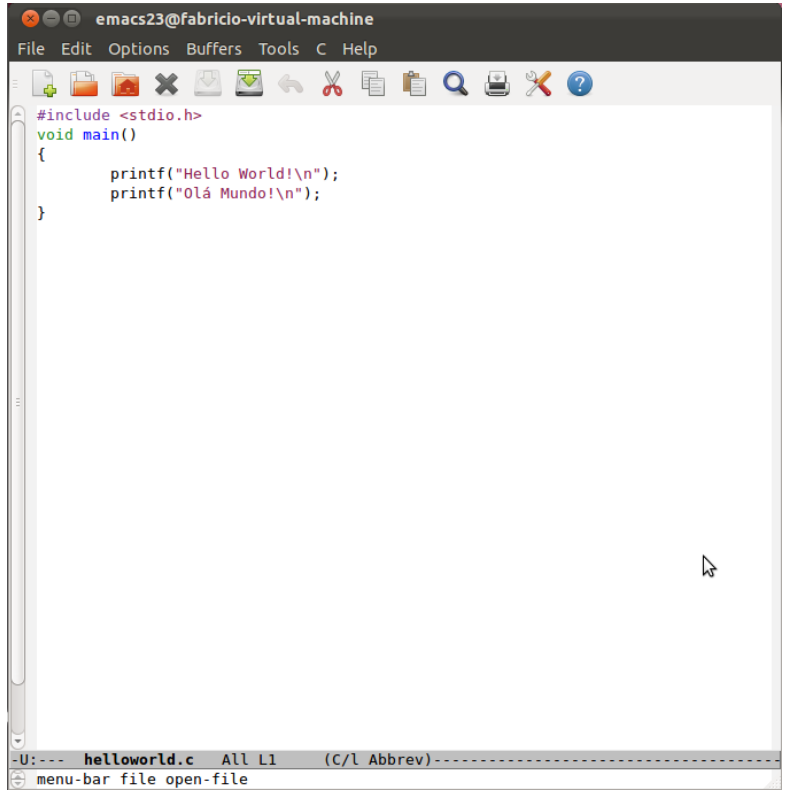




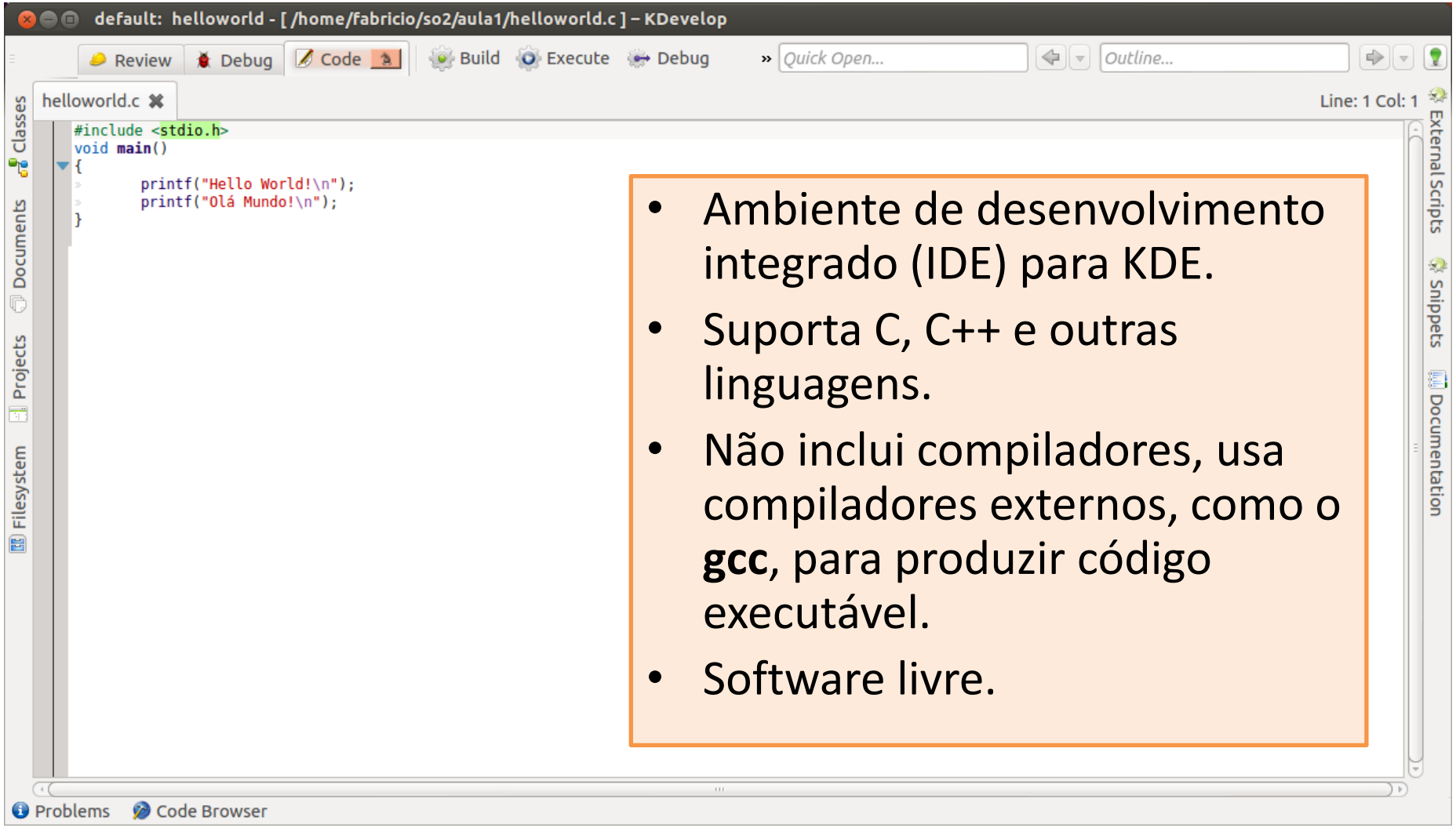
Editores de Texto

- Modo Gráfico
  - **gedit**
    - Editor padrão da interface Gnome
  - **Kate**
    - Editor padrão da interface KDE
  - **Emacs**
    - Bastante poderoso e personalizável



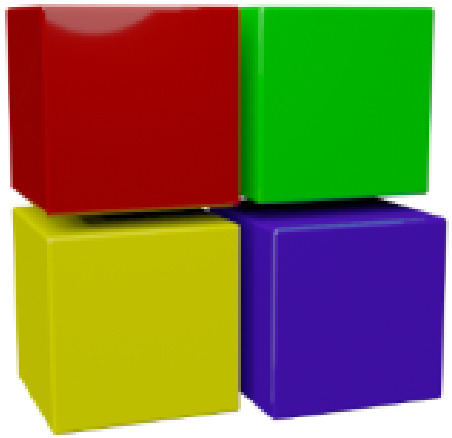


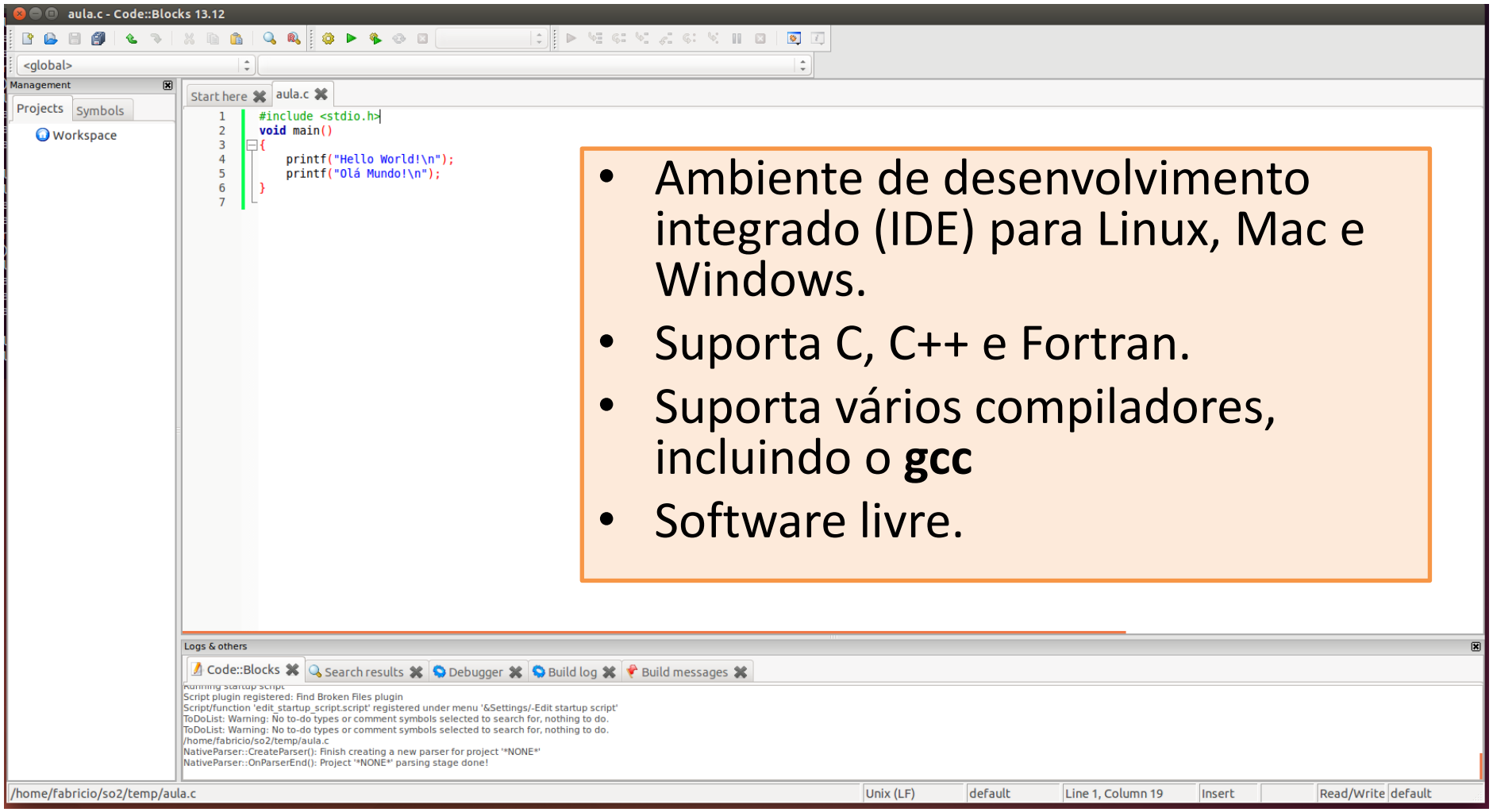
KDevelop



- Ambiente de desenvolvimento integrado (IDE) para KDE.
- Suporta C, C++ e outras linguagens.
- Não inclui compiladores, usa compiladores externos, como o **gcc**, para produzir código executável.
- Software livre.

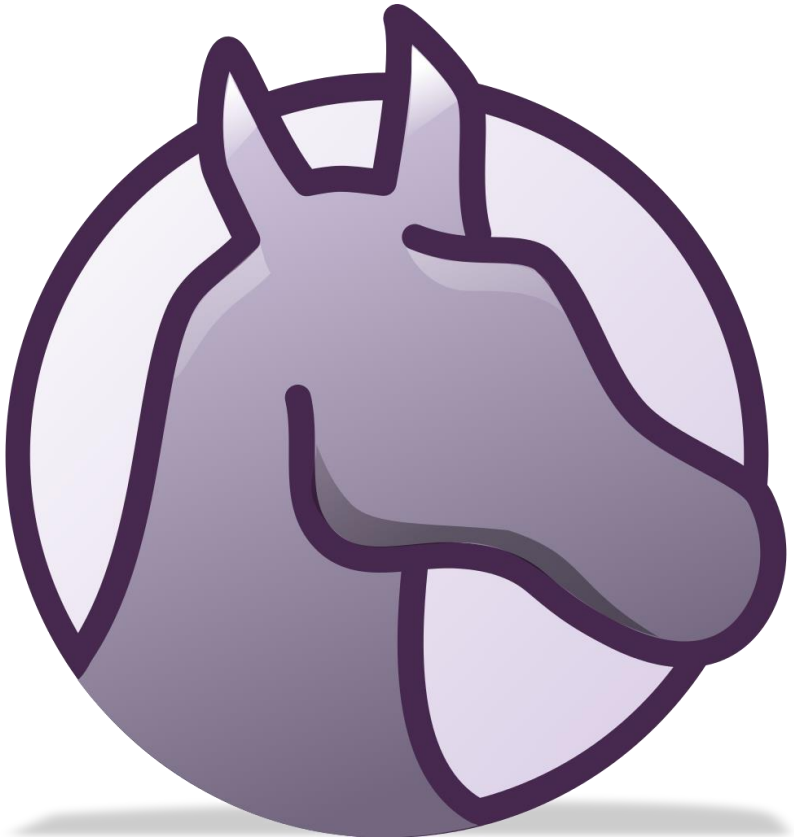
Code::Blocks

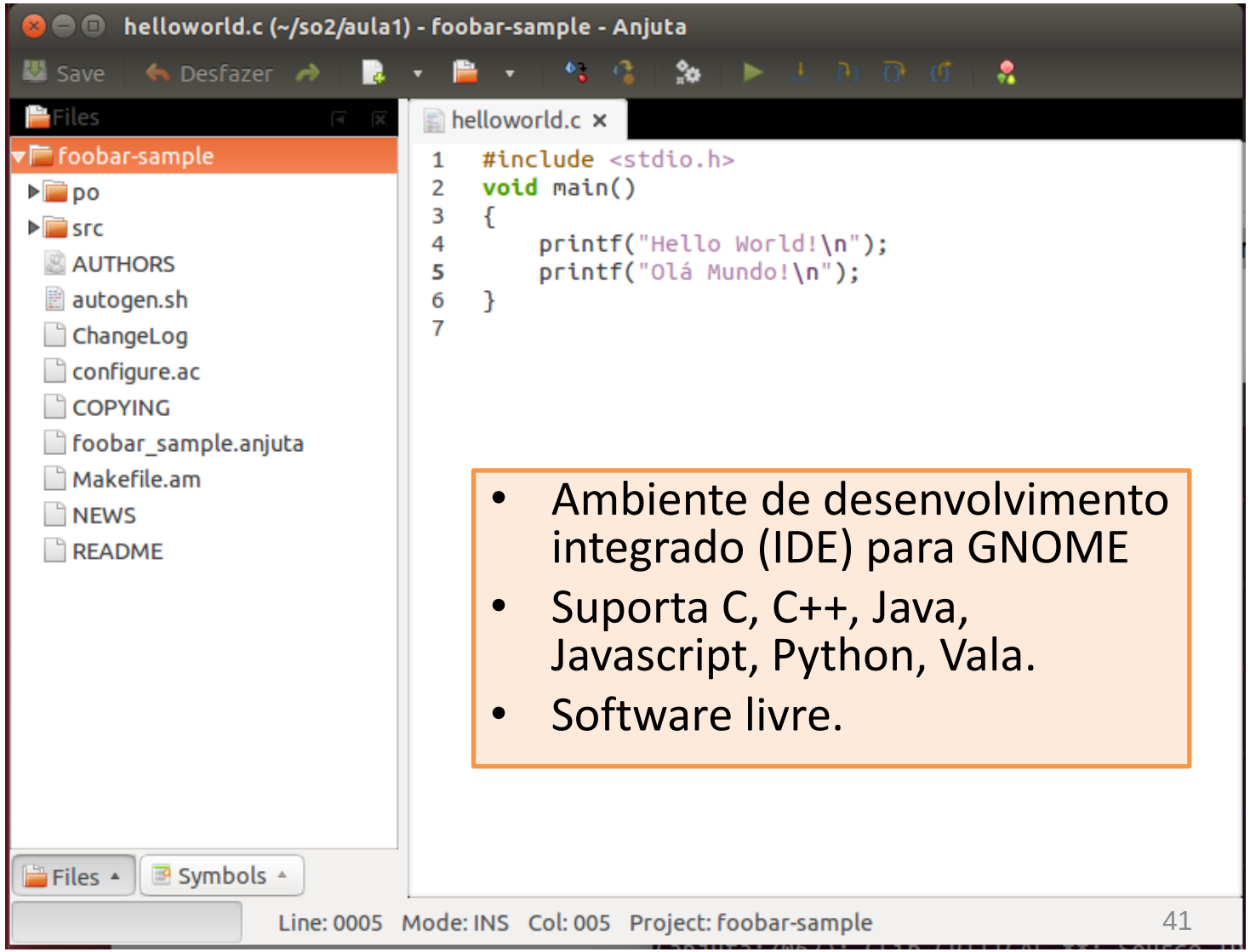




- Ambiente de desenvolvimento integrado (IDE) para Linux, Mac e Windows.
- Suporta C, C++ e Fortran.
- Suporta vários compiladores, incluindo o **gcc**
- Software livre.

Anjuta DevStudio



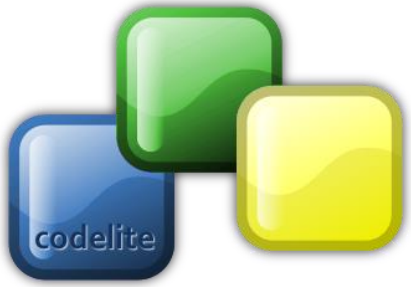


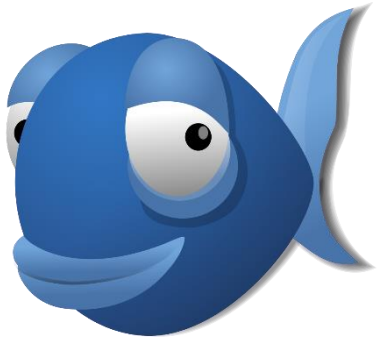
- Ambiente de desenvolvimento integrado (IDE) para GNOME
- Suporta C, C++, Java, Javascript, Python, Vala.
- Software livre.

Vídeo mostrando editores de texto no Linux Ubuntu: vi, nano, gedit: https://youtu.be/69Cdl7oScoc

Outros editores e ambientes de desenvolvimento

- Netbeans
- Eclipse
- CodeLite IDE
- Atom
- Sublime Text
- Bluefish Editor
- Brackets
- Geany IDE





Programando

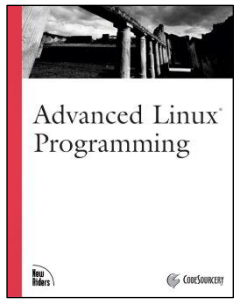
- Use seu editor de textos preferido para criar código-fonte na linguagem:
  - C
    - *.c* ou *.h*
  - C++
    - *.cpp, .hpp, .cxx, .hxx, .C, .H*
  - *Etc...*

Compiladores

- GNU Compiler Collection (GCC)
  - C, C++, Java, Objective-C, Fortran, Ada, Go
  - `gcc helloworld.c`
    - Compila e linka, gerando executável
  - `gcc -c main.c`
    - Apenas compila código C, gerando arquivo objeto
  - `g++ -c reciprocal.cpp`
    - Apenas compila código C++, gerando arquivo objeto
  - `g++ -c -I ../include reciprocal.cpp`
    - Instrui o compilador a procurar cabeçalhos também em ../include

Códigos-Fontes de Exemplo nas Aulas

- São códigos de exemplo do livro *Advanced Linux Programming*
  - Livro publicado sob a Open Publication License
  - Códigos-Fontes cobertos pela GNU General Public License
    - Livro em PDF e códigos disponíveis para *download* no *Google Classroom*
  - Algumas atualizações foram feitas para que os códigos compilem e executem corretamente nos sistemas atuais
    - Tais mudanças estão apontadas nos *slides* e **não estão incluídas** nos arquivos fornecidos.
  - Tratamentos de exceções, teste de consistência de entradas, etc. são omitidos, mantendo no código apenas o essencial para demonstrar as funcionalidades, para fins didáticos e fácil visualização nos slides.
    - **Não faça isso em seus códigos!**



main.c

```c
#include <stdio.h>
#include <stdlib.h>
#include "reciprocal.hpp"
int main  (int  argc, char **argv)
{
        int i;
        i =  atoi (argv[1]);
        printf ("O inverso de %d é %g\n", i, reciprocal (i));
        return 0;
}
```

reciprocal.cpp

```c
#include <cassert>
#include "reciprocal.hpp"
double reciprocal  (int i)
{
        // deve ser diferente de zero
  assert (i !=  0);
        return 1.0/i;
}
```

reciprocal.hpp

```c
#ifdef __cplusplus
extern "C"  {
#endif

extern double reciprocal(int i);

#ifdef __cplusplus
}
#endif
```

Compiladores

- `g++ -c -O2 reciprocal.cpp`
  - Compila com nível de otimização 2 (para código de produção)
- `g++ -o reciprocal main.o reciprocal.o`
  - “Linka” os objetos .o gerando o arquivo executável reciprocal
- `./reciprocal 7`
  - Executa o programa compilado, passando 7 como argumento

Compiladores

- Você pode criar um arquivo `Makefile` para indicar dependências e automatizar a compilação com **`make`**

```
reciprocal:  main.o reciprocal.o
        g++  $(CFLAGS) -o reciprocal  main.o reciprocal.o

main.o: main.c reciprocal.hpp
        gcc  $(CFLAGS) -c main.c

reciprocal.o:  reciprocal.cpp reciprocal.hpp
        g++  $(CFLAGS) -c reciprocal.cpp

clean:
        rm  -f *.o reciprocal
```

Compiladores

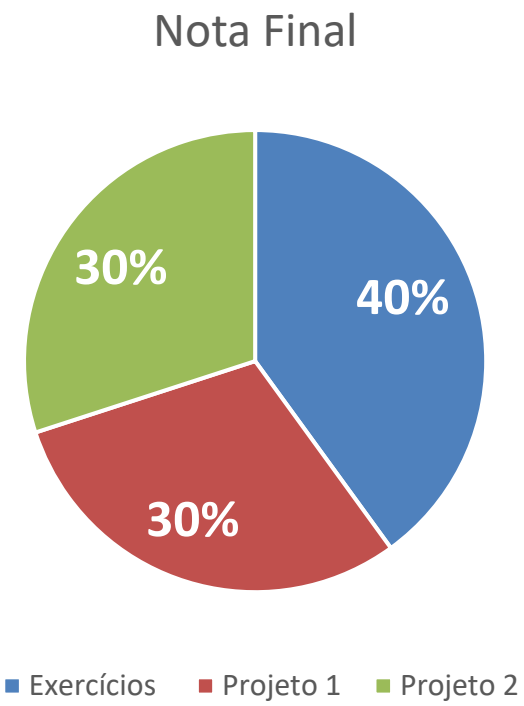
- `make`
  - Checa dependências em Makefile, compila códigos, gera objetos, faz ligações e gera executável
- `make clean`
  - Remove os arquivos objetos e executável, mantendo apenas os códigos-fontes
- `make CFLAGS=-O2`
  - Passa a *flag* para o compilador, gerando executável otimizado

Programação da Disciplina

- Introdução
- Técnicas Básicas
- Processos
- Threads
- Comunicação entre Processos
- Sistema de Arquivos
- Dispositivos
- O sistema `/proc`
- Chamadas de Sistema do Linux

Método de Avaliação

- Exercícios – **40%**
  - 8 exercícios – **5%** cada
  - Implementações relacionadas aos temas das aulas
- Projetos – **60%**
  - Interpretador de Comandos – **30%**
  - Simulador de Sistema de Arquivos – **30%**



Bibliografia

1. NEMETH, Evi.; SNYDER, Garth; HEIN, Trent R.;. *Manual Completo do Linux: Guia do Administrador*. São Paulo: Pearson Prentice Hall, 2007. Cap. 1.
2. DEITEL, H. M.; DEITEL, P. J.; CHOFFNES, D. R.; *Sistemas Operacionais: terceira edição*. São Paulo: Pearson Prentice Hall, 2005. Cap. 20.
3. MITCHELL, Mark; OLDHAM, Jeffrey; SAMUEL, Alex; *Advanced Linux Programming*. New Riders Publishing: 2001. Cap. 1.
4. TANENBAUM, Andrew S.; BOS, Herbert. *Sistemas Operacionais Modernos*. 4ed. São Paulo: Pearson Education do Brasil, 2016. Cap. 10.