MC102—Algoritmos e Programação de Computadores

Uso dos Laboratórios de Informática do Ciclo Básico II

Acessando os computadores

Login: Para acessar os computadores, você deve utilizar um nome usuário e senha fornecidos pela administração dos laboratórios e é independente das demais senhas que você possui na UNICAMP.

Para mudar a sua senha, digite `kpasswd` em um terminal e siga as instruções do programa (ele pedirá para que você digite a sua senha atual, uma senha nova e confirme a nova senha). Tenha bastante cuidado com a senha pois é difícil recuperar uma senha perdida nos laboratórios do PB.

Abrindo o terminal: No canto superior esquerdo da tela existem três menus. Clique sobre o menu "Applications", depois em "System Tools" e por fim em "Terminal". Aparecerá uma tela com apenas uma linha:

```
[ra009702@catatau] ~$
```

Os passos necessários para se abrir um terminal podem variar para alguns alunos, dependendo da configuração dos computadores, mas independente disso há sempre uma forma de abrir o Terminal.

Essa tela serve para digitar comandos para o sistema operacional. Após a digitação de um comando, você deve pressionar a tecla enter para que esse comando seja executado.

Tipicamente, existem ícones para realizar todas as funções explicadas nesse tutorial, mas sua localização pode mudar dependendo da configuração de cada máquina, assim, esse roteiro de uso dos laboratórios não irá abordar esses atalhos, somente comandos digitados no terminal.

Acessando a Internet

Para acessar uma página da Internet, digite no terminal o comando `firefox` seguido de um e comercial (&), como no exemplo abaixo:

```
[ra009702@catatau] ~$ firefox &
```

Surgirá uma tela semelhante a aquelas que vocês utilizam para navegar nos demais laboratórios da UNICAMP, ou mesmo nas suas casas.

Estrutura de Diretórios e alguns comandos Linux

A estrutura de diretórios do Linux é um pouco diferente do Windows. No Linux temos um diretório principal, chamado de raiz do sistema, a partir do qual todos os arquivos podem ser acessados, independente se ele está em um disco rígido, CD, disquete ou em um outro computador ligado a rede. Nessa seção mostraremos os principais diretórios de um sistema Linux típico e alguns comandos para manipular arquivos e diretórios.

Estrutura de diretórios no Linux

/: : Raiz do sistema.

/home: Diretórios e arquivos de trabalho dos usuários comuns do sistema. É o equivalente a pasta "Documents and Settings" do Windows. Nos laboratórios do PB, todo arquivo gravado nesse diretório será apagado quando a máquina for desligada, assim, tome bastante cuidado ao gravar algo aqui.

/tmp: Diretório reservado para os arquivos temporários utilizados por algumas aplicações. Você pode gravar seus programas nessa pasta, pois ela não é apagada quando a máquina é desligada

/mnt: Diretório reservado para o acesso a outros dispositivos de armazenamento, como disquetes e CD-ROMs. Como não é possível gravar arquivos nas pastas pessoais, recomendamos que todo aluno traga um disquete para gravar seu trabalho. Outra alternativa é enviar os arquivos para seus e-mails pessoais periodicamente.

Gerenciamento de Arquivos e diretórios no Linux

Veremos agora alguns comandos básicos do sistema Linux que permitem manipular arquivos e diretórios. Todos esses comandos devem ser digitados no terminal, como visto anteriormente.

- Comandos que manipulam diretórios

  mkdir <nome_do_diretorio>: cria um diretório no sistema.
  Ex: `mkdir minha_pasta`

  rmdir <nome_do_diretorio>: remove um diretório vazio do sistema.
  Ex: `rmdir minha_pasta`

  cd <nome_do_diretorio>: muda de diretório. Ex:
  `cd minha_pasta` Vai para a pasta minha pasta
  `cd ..` Vai para a pasta pai

  pwd: indica o nome do diretório atual.

- Comandos que manipulam arquivos

  ls: lista todos os arquivos e subdiretórios do diretório corrente.

  mv <origem> <destino>: move o arquivo origem para o arquivo destino. Ex:
  `mv primeiro.c segundo.c` move o arquivo primeiro.c para segundo.c
  `mv primeiro.c /tmp` move o arquivo primeiro.c para a pasta /tmp

  cp <origem> <destino>: copia o arquivo origem para o arquivo destino. Ex:
  `cp primeiro.c segundo.c` copia o arquivo primeiro.c para segundo.c
  `cp primeiro.c /tmp` copia o arquivo primeiro.c para a pasta /tmp

  rm <arquivo>: apaga um arquivo. Ex: `rm primeiro.c`

Criando e alterando programas

No Linux, existem diversos editores de texto, alguns em modo texto e outros gráficos, com recursos que auxiliam no desenvolvimento de programas. Recomendamos o uso do Kate ou do Gedit como editor de texto, pois ambos são gráficos, os comandos são intuitivos (tudo que você quiser fazer nele pode ser feito utilizando o menu) e auxiliam no desenvolvimento de programas em linguagem C, a linguagem de programaçã que será utilizada durante toda a disciplina. Para acessar o Kate, digite no terminal o comando `kate` seguido de um e comercial (&), como no exemplo abaixo:

```
[ra009702@catatau] ~$ kate &
```

Para acessar o Gedit, digite no terminal o comando `gedit` seguido de um e comercial (&), como no exemplo abaixo:

```
[ra009702@catatau] ~$ gedit &
```

As funções mais imporantes do Kate e do Gedit estão no menu arquivo:

Novo: Cria um novo arquivo de texto.

Abrir: Abre um arquivo já existente.

Salvar: Salva o arquivo atual. Se ele ainda não tiver um nome, o programa pedirá a você que escolha um nome para ele. Programas em linguagem C costumam ter a extensão `.c`, por exemplo: *fonte.c*, *exemplo.c*, *aio.c*.

Dependendo da configuração de cada máquina, os nomes dos menus podem estar em inglês.

Para testarmos as funcionalidades dos editores de texto, abra um deles, digite o seguinte texto e salve-o com o nome hello.c na pasta /tmp. Esse será o primeiro código-fonte em linguagem C que vocês verão e a função dele é imprimir a expressão "Hello World!!!" na tela. Os detalhes desse programa serão explicados nas aulas iniciais.

```
#include <stdio.h>
main () {
        printf ("Hello World!!!");
}
```

Compilando programas

A tarefa de transformar um código-fonte (que é um arquivo texto) em um programa executável é chamada de compilação. No Linux, o compilador utilizado é o `gcc`, cuja sintaxe é:

```
gcc <nome do código fonte> -o <nome do executável>
```

Por exemplo, o comando

```
gcc exemplo.c -o exemplo
```

realizará a compilação do arquivo exemplo.c gerando o executável exemplo. Para executar um programa que acabamos de compilar, devemos digitar:

```
./<nome do programa>
```

assim, para executar o programa `exemplo` criado com o comando anterior, devemos digitar

```
./exemplo
```

O próximo passo agora é testarmos a compilação do programa `hello`, cujo código-fonte foi escrito na etapa anterior. Para compilarmos o programa, digite o seguinte no seu terminal:

```
cd /tmp
gcc hello.c -o hello
```

e para executá-lo, digite:

```
./exemplo
```