

Aula Prática 1

Objetivos:

- Uso das funções *printf* (incluindo as cores da tabela ANSI), da palavra-chave *sizeof* e do operador ternário.
- Exibir os tamanhos dos tipos básicos, dos modificadores de sinal, dos modificadores de largura e das possíveis combinações destes 3 grupos.
- Uso de arquivos de dependências específicos para os sistemas operacionais das famílias *BSD* e *Linux*.
- Uso de um sistema de controle de versão (RCS - *Revision Control System*).

Versão Inicial: 16/04/2024

Prazo: 19/04/2024 – 08:00

Observações:

- Leia este enunciado com **MUITA** atenção até o final antes de iniciar o trabalho.
- Os arquivos solicitados deverão estar disponíveis nos diretórios correspondentes (*Aulas-Praticas* e *RCS*) até o prazo estipulado acima. Cuidado com os nomes dos diretórios e dos arquivos. Deverão ser exatamente os definidos neste roteiro (incluindo maiúsculas, minúsculas, caracteres especiais e extensões, se existentes).
- **As tarefas deverão ser executadas na ordem solicitada neste roteiro.**
- O padrão de nomenclatura definido em aula para os identificadores deverá ser utilizado:
 - **Snake Case**
 - letras maiúsculas, dígitos e o caractere sublinhado;
 - macros e constantes;
 - Exemplo: **COMPRIMENTO_MAXIMO_NOME**
 - **Camel Case**
 - letras minúsculas, com exceção do primeiro caractere de cada palavra e a partir da segunda palavra que compõe o identificador;
 - tipos, membros de tipos enumerados e variáveis;
 - Exemplo: **comprimentoNomeCompleto**
 - **Pascal Case**
 - letras minúsculas, com exceção do primeiro caractere de cada palavra que compõe o identificador;
 - funções
 - Exemplo: **CalcularFatorial**
 - Todos os identificadores podem conter dígitos, mas nenhum identificador pode começar com dígito.
 - Todos os identificadores devem ser significativos e sem abreviações.
 - Identificadores de função devem começar com um verbo no infinitivo.
- Os arquivos de dependências deverão possibilitar que a compilação e que a *linkedição* sejam executadas utilizando-se tanto o *gcc*, quanto o *clang*. A seleção da ferramenta utilizada deverá ser realizada no momento da execução do comando *make*. O *gcc* deverá ser considerado como o valor padrão para a ferramenta de compilação e de *linkedição*.

Para a definição da ferramenta desejada, deverá ser utilizada uma macro no *FreeBSD* e um argumento com o valor desejado no *Linux*.

As duas macros utilizadas deverão ser **GCC** e **CLANG** (definidas usando a opção de linha de comando **-D** do comando **make**).

O argumento, identificado por **cc**, deverá ser igual a **GCC** ou a **CLANG**.

- Independente da ferramenta utilizada para a compilação, as opções de compilação poderão ser redefinidas no instante da execução do comando **make** (mantendo-se a exibição de todas as mensagens de advertência, definida pelo valor **-Wall**). O valor padrão para estas opções deverá ser **-Wall -ansi**.

Estas opções poderão ser redefinidas através de macros ou através de argumentos (de forma semelhante àquela utilizada para definir o compilador/linkeditor).

No *FreeBSD* deverão ser definidas as macros **ANSI**, **C89**, **C90**, **C99** e **C11**.

No *Linux* deverá ser definido o argumento **dialeto** com um dos seguintes valores: **ANSI**, **C89**, **C90**, **C99** ou **C11**.

- Os arquivos de dependências deverão incluir a macro **DIALECT** contendo o dialeto a ser utilizado na compilação do código. Esta macro será inicialmente igual a **ansi** e poderá ser alterada para **c89**, **c90**, **c99** ou **c11** de acordo com o esquema definido acima.
- Os arquivos de dependências deverão incluir também a macro **STANDARD** contendo a opção de linha de comando correspondente ao dialeto selecionado. Se, por exemplo, o dialeto selecionado for o **ANSI**, esta macro deverá ser igual a **-ansi**. Por outro lado, se o dialeto for uma das outras quatro opções, esta macro deverá ser igual a **-std=CXX**, onde **XX** deverá ser substituído pelo número correspondente (se o dialeto for igual a **C89**, **XX** deverá ser igual a **89**, se o dialeto for igual a **C90**, **XX** deverá ser igual a **90** e assim por diante).
- A *linkedição* deverá utilizar a opção **-Wall**.
- Cuidado com os nomes das macros e dos rótulos. Deverão ser exatamente os definidos neste roteiro (maiúsculas, minúsculas, caracteres especiais e extensões, se existentes).
- Todos os rótulos solicitados no roteiro são obrigatórios. Durante a correção, caso não seja possível alcançar os objetivos (binários e/ou bibliotecas e limpezas de código) solicitados, a nota correspondente ao item/aula em questão será igual a zero.
- Seguem alguns exemplos (todos devem funcionar):
 - **make** - compila/linkedita (tanto no *FreeBSD*, quanto no *Linux*) com a ferramenta e dialeto padrões, ou seja, **gcc** e **ANSI** respectivamente.
 - **make clean-all all**
 - **make clean-all aula01**
 - **make clean aula0101**
 - **make -DGCC** - compila/linkedita usando o **gcc** e o dialeto **ANSI** (somente *FreeBSD*).
 - **make -DCLANG** - compila/linkedita usando o **clang** e o dialeto **ANSI** (somente *FreeBSD*).
 - **make cc=GCC** - compila/linkedita usando o **gcc** e o dialeto **ANSI** (somente *Linux*).
 - **make cc=CLANG** - compila/linkedita usando o **clang** e o dialeto **ANSI** (somente *Linux*).
 - **make -DCLANG -DC89** - compila/linkedita usando o **clang** e o dialeto **C89** (somente *FreeBSD*).
 - **make -DCLANG -DC11** - compila/linkedita usando o **clang** e o dialeto **C11** (somente *FreeBSD*).
 - **make cc=CLANG dialeto=C99** - compila/linkedita usando o **clang** e o dialeto **C99** (somente *Linux*).
 - **make cc=GCC dialeto=C90** - compila/linkedita usando o **gcc** e o dialeto **ANSI** (somente *Linux*).

- Inclua, no início de todos os arquivos solicitados (código-fonte e arquivos de dependências), os seguintes comentários (**sem caracteres especiais**):

```
Universidade Federal do Rio de Janeiro
Escola Politecnica
Departamento de Eletronica e de Computacao
EEL270 - Computacao II - Turma 2023/2
Prof. Marcelo Luiz Drumond Lanza
Autor: <nome completo>
Descricao: <descricao sucinta dos objetivos do programa>
$Author$
$Date$
$Log$
```

- Inclua, no final de todos os arquivos solicitados, o seguinte comentário:

```
$RCSfile$
```

Antes de começar as tarefas desta aula, leia com atenção os itens 1, 2, e 3.

- Abra um terminal gráfico ou texto, execute o comando **screen** e crie 3 terminais virtuais (**CTRL+a c**):

- Edição
- Compilação, *linkedição* e testes usando o sistema operacional *Linux*.
- Compilação, *linkedição* e testes usando o sistema operacional *FreeBSD*.

A partir de qualquer máquina é possível se conectar remotamente a outra máquina executando o comando:

```
"ssh nome-da-maquina" (sem as aspas)
```

Para se conectar à rede DEL, usando o *MobaXterm* ou o cliente *SSH* desejado, é preciso se conectar primeiro à máquina “*loghost02.del.ufrj.br*”. A partir desta máquina é possível se conectar às máquinas dos laboratórios executando o comando:

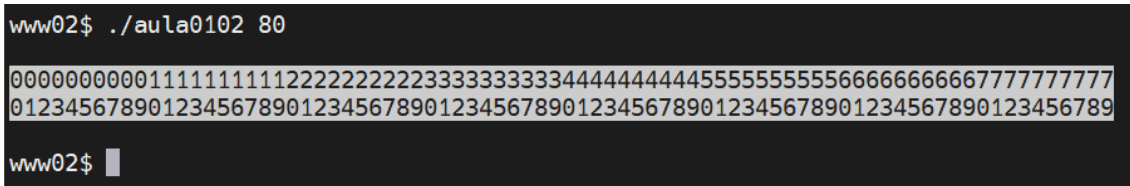
```
"ssh ligXYZ" (sem as aspas)
```

onde **XYZ** pode algum valor nas faixas 025-044 e 073-102.

- Crie o diretório “*~/private/EEL270/2024-1/Aulas-Praticas/RCS*”.
- Copie o arquivo **cores.h** do diretório *~marcelo.lanza/public/EEL270/2024-1/Aulas-Teoricas* para o diretório *~/private/EEL270/2024-1/Aulas-Praticas*. Este arquivo contém macros correspondendo às sequências de escape ANSI para as 8 cores básicas e variações. Estas sequências poderão ser utilizadas na função *printf* para trocar a cor de fundo e/ou a cor dos caracteres.
- Crie o arquivo **aula0101.h** contendo a definição dos tipos **us** (correspondendo a **short unsigned**) e **ul** (correspondendo a **long unsigned**) e a definição do protótipo da função **ExibirCabecalho**. Esta função deverá receber um inteiro não negativo correspondendo ao número de colunas desejado. A função deverá exibir um cabeçalho no formato mostrado na figura 1 (neste exemplo o número de colunas é igual a 80). Note que o fundo deverá ser branco e que os caracteres deverão ser pretos.

A macro referente à combinação *ifndef* e *define* deverá ser igual a *AULA0101* e deverá ser definida como uma *string* igual a "*aula0101.h \$Revision\$*".

```
void
ExibirCabecalho (us);
```



The image shows a terminal window with a black background. The prompt is 'www02\$'. The command entered is './aula0102 80'. The output is a long string of characters: '0000000000111111111222222222233333333334444444444555555555566666666667777777777' followed by '01234567890123456789012345678901234567890123456789012345678901234567890123456789'. The prompt 'www02\$' is shown again at the bottom.

Figura 1

5. Crie o arquivo *aula0101.c* contendo o código-fonte da função *ExibirCabecalho*.
6. Crie o arquivo *aula0102.c* contendo o código-fonte de um programa de testes para a função *ExibirCabecalho*. Este programa deverá receber, através de um argumento de linha de comando, o número de colunas desejado, que deverá ser um valor entre 1 e 500. A função *strtol* deverá ser utilizada para converter o argumento de linha de comando (do tipo *string*) para inteiro (do tipo *us*). O programa deverá incluir todas as validações (possíveis e necessárias) do argumento recebido. Todas as mensagens de erro deverão ser exibidas em vermelho.
7. Crie os arquivos de dependências (*BSDmakefile* e *GNUMakefile*) contendo as macros *CC*, *LD*, *CFLAGS*, *LFLAGS*, *DIALECT*, *STANDARD*, *AULA01*, *AULA0102OBS*, *EXECS*, *LIBS* e *ALL*.

A macro *STANDARD* deverá receber o valor padrão *-ansi*. Este valor poderá ser alterado se durante a execução do comando *make*, o dialeto for alterado (conforme definido das observações iniciais). Neste caso, a macro deverá receber um dos seguintes valores:

-std=c89, *-std=c90*, *-std=c99* ou *-std=c11*.

A macro *AULA01* deverá corresponder ao valor *aula0102*, enquanto que a macro *AULA0102OBS* deverá corresponder aos valores *aula0101.o* e *aula0102.o*. A macro *EXECS* deverá corresponder ao valor da macro *AULA01*, enquanto que a macro *LIBS* deverá corresponder a uma *string* vazia. Finalmente, a macro *ALL* deverá corresponder aos valores das macros *EXECS* e *LIBS*.

Inclua, nos arquivos de dependências, os rótulos *all*, *aula01*, *aula0102*, *clean-all*, *clean*, *clean-objs*, *clean-bsd*, *clean-linux*, *clean-gcc*, *clean-clang*, *clean-ansi*, *clean-c89*, *clean-c90*, *clean-c99* e *clean-c11* com as declarações necessárias.

O objetivo *aula01* deverá permitir gerar todos os binários solicitados nesta aula.

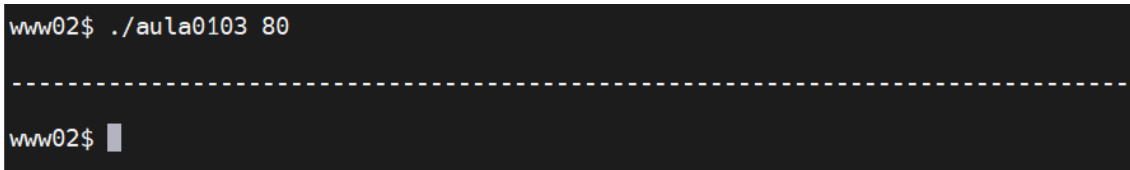
Inclua, nos comandos relativos ao objetivo *aula0102*, o comando necessário para criar uma cópia do binário com o nome que inclua o sistema operacional, a ferramenta de compilação/linkedição e o dialeto da linguagem C utilizado, de acordo com o exemplo abaixo (informações na ordem mostrada no exemplo):

aula0102-FreeBSD-gcc-ansi - versão gerada no *FreeBSD* usando o *gcc* e o padrão *ANSI*.

8. Gere e teste as 20 possíveis versões do binário *aula0102* (*Linux* x *FreeBSD*, *clang* x *gcc*, *ANSI* x *C89* x *C90* x *C99* x *C11*).

9. Crie uma cópia do arquivo *aula0102.c* com o nome *aula0103.c*.
10. Submeta os arquivos *aula0101.h*, *aula0101.c*, *aula0102.c*, *BSDmakefile* e *GNUMakefile* ao sistema de controle de versão.
11. Recupere uma cópia de leitura do arquivo *aula0102.c* e uma cópia de escrita dos demais arquivos.
12. Inclua no arquivo *aula0101.h* o protótipo da função *ExibirHifens*. Esta função deverá receber um inteiro não negativo correspondendo ao número de hifens desejado. A função deverá exibir uma linha contendo os hifens desejados no formato mostrado na figura 2 (neste exemplo o número de hifens é igual a 80). Note que deverá ser incluída uma linha em branco antes e uma linha em branco depois da linha contendo os hifens. Nesta função o fundo deverá ser preto, enquanto que os caracteres deverão ser brancos.

```
void
ExibirHifens (us);
```



```
www02$ ./aula0103 80
-----
www02$ █
```

Figura 2

13. Inclua, no arquivo *aula0101.c*, o código-fonte da função *ExibirHifens*.
14. Altere o código-fonte contido no arquivo *aula0103.c* para criar um programa de testes para a função *ExibirHifens*. Este programa deverá receber, através de um argumento de linha de comando, o número de hifens desejado, que deverá ser um valor entre 1 e 500. A função *strtol* deverá ser utilizada para converter o argumento de linha de comando (do tipo *string*) para inteiro (do tipo *us*). O programa deverá incluir todas as validações (possíveis e necessárias) do argumento recebido.
15. Inclua, no arquivo de dependências, a macro *AULA0103OBS* e o rótulo *aula0103* com as declarações necessárias. Atualize os valores das macros, onde for necessário.
16. Gere e teste as 20 possíveis versões do executável *aula0103*.
17. Submeta os arquivos *aula0101.h*, *aula0101.c*, *aula0103.c* e os arquivos de dependências ao sistema de controle de versão.
18. Recupere uma cópia de leitura dos arquivos *aula0101.h*, *aula0101.c*, *aula0103.c* e uma cópia de escrita dos arquivos de dependências.
19. Crie o arquivo *aula0104.c* contendo o código-fonte de um programa que exiba as 256 possíveis cores de um terminal. A saída gerada pelo programa deverá ser exatamente igual à mostrada na figura 3.
 - Para exibir um texto com uma das 256 possíveis cores, utilize a sequência de escape:

```
"\e[38;5;cor m"
```

onde cor deverá ser substituída por um dos 256 possíveis valores.

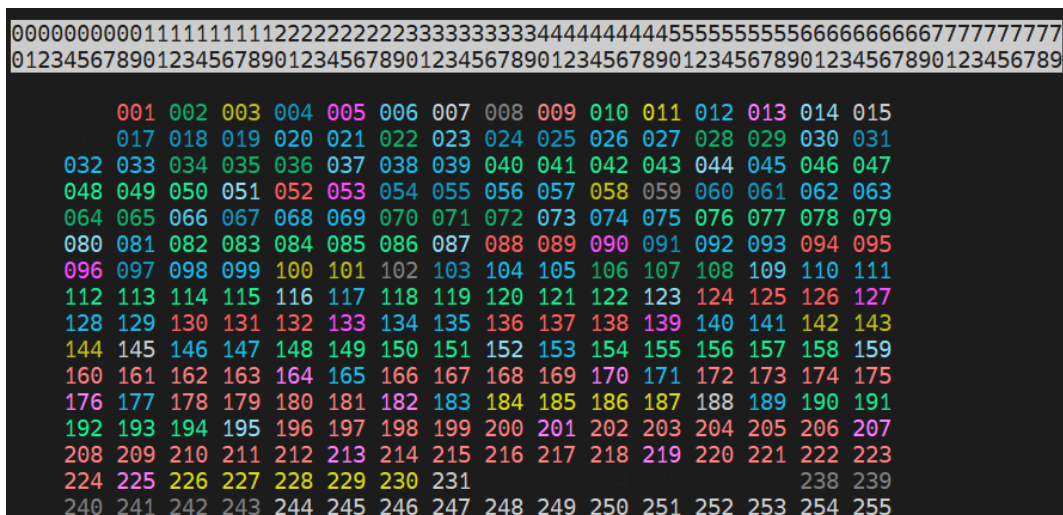


Figura 3

20. Inclua, no arquivo de dependências, a macro **AULA0104OBS** e o rótulo **aula0104** com as declarações necessárias. Atualize os valores das macros, onde for necessário.
21. Gere e teste as 20 possíveis versões do executável **aula0104**.
22. Submeta o arquivo **aula0104.c** e os arquivos de dependências ao sistema de controle de versão.
23. Recupere uma cópia de leitura do arquivo **aula0104.c** e uma cópia de escrita dos arquivos de dependências.
24. Crie o arquivo **aula0105.c** contendo o código-fonte de um programa que receba um nome que possa conter caracteres de espaço e que tenha no máximo **50** caracteres.

Este nome poderá ser recebido através de **um único argumento** de linha de comando ou através de **vários argumentos** de linha de comando. O programa deverá exibir o nome recebido delimitado por aspas e centralizado na tela do terminal (assuma que a tela do terminal tem **100** colunas).

O formato de exibição deverá ser o mostrado na figura 4, incluindo os números das colunas:

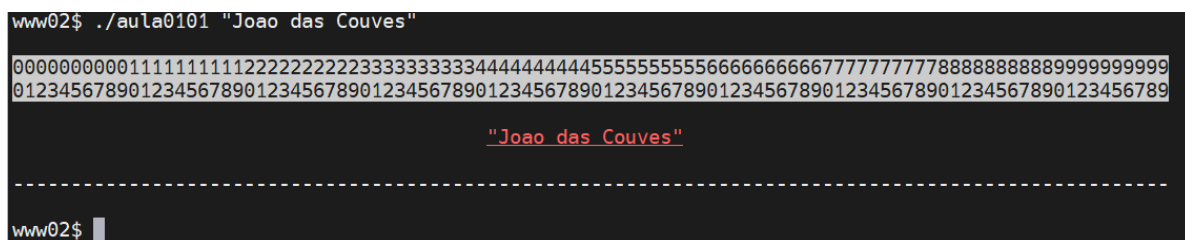


Figura 4

As funções **ExibirCabecalho** e **ExibirHifens** deverão ser utilizadas para exibir os números e os hifens. O nome deverá ser exibido vim a com texto em vermelho e sublinhado.

Note que existe uma linha em branco antes e após o cabeçalho.

A função **strlen** retorna o comprimento útil de uma **string**, ou seja, o comprimento da **string** não incluindo o caractere de final de **string** (**EOS** – **End of String**).

25. Inclua, no arquivo de dependências, a macro **AULA0105OBS** e o rótulo **aula0105** com as declarações necessárias. Atualize os valores das macros, onde for necessário.

26. Gere e teste as 20 possíveis versões do executável *aula0105*.
27. Submeta o arquivo *aula0105.c* e os arquivos de dependências ao sistema de controle de versão.
28. Recupere uma cópia de leitura do arquivo *aula0105.c* e uma cópia de escrita dos arquivos de dependências.
29. Crie o arquivo *aula0106.c* contendo um programa que exiba os tamanhos dos 5 tipos básicos, dos 2 modificadores de sinal e dos 3 modificadores de largura.

Use o formato mostrado na figura 5.

Exiba os títulos correspondentes a cada bloco de informação (sem caracteres especiais):

Tamanhos dos Tipos Basicos

Tamanhos dos Modificadores de Sinal

Tamanhos dos Modificadores de Largura

Estes títulos deverão ser exibidos de forma centralizada e deverão estar sublinhados, usando branco. Antes do título, exiba uma linha em branco, uma linha contendo 100 hifens e uma linha em branco (usando a função *ExibirHifens*).

Após o título inclua uma linha em branco e a seguir os tamanhos dos tipos desejados.

O nome do tipo (exatamente igual à palavra-chave correspondente) deverá ser exibido a partir da coluna 1 usando a cor verde, enquanto que o tamanho em bytes deste tipo (utilizando-se a palavra-chave *sizeof*) deverá ser exibido na mesma linha a partir da coluna 91. A palavra *byte* ou a palavra *bytes* (mantendo a concordância) deverá ser exibida após um caractere de espaço colocado após o tamanho do tipo em questão. O tamanho e a palavra *byte/bytes* deverão ser exibido usando a cor amarela.

Após o último tipo, deverá ser incluída uma linha em branco, uma linha com 100 hifens e uma linha em branco (usando a função *ExibirHifens*).

O **argumento** da função *printf* deverá usar o operador ternário para selecionar entre a palavra *byte* e a palavra *bytes*.

Operador Ternário:

Condição ? verdadeiro : falso

Onde:

Condição corresponde à condição que será testada.

verdadeiro corresponde à ação que será executada quando a condição for verdadeira.

falso corresponde à ação que será executada quando a condição for falsa.

30. Inclua, no arquivo de dependências, a macro *AULA0106OBS* e o rótulo *aula0106* com as declarações necessárias. Atualize os valores das macros, onde for necessário.
31. Gere e teste as 20 possíveis versões do executável *aula0106*.
32. Crie uma cópia do arquivo *aula0106.c* com o nome *aula0107.c*.

33. Submeta o arquivo ***aula0106.c*** e os arquivos de dependências ao sistema de controle de versão.
34. Recupere uma cópia de leitura do arquivo ***aula0106.c*** e uma cópia de escrita dos arquivos de dependências.

```
www02$ ./aula0106

-----

Tamanhos dos Tipos Basicos

void: 1 byte
char: 1 byte
int: 4 bytes
float: 4 bytes
double: 8 bytes

-----

Tamanhos dos Modificadores de Sinal

signed: 4 bytes
unsigned: 4 bytes

-----

Tamanhos dos Modificadores de Largura

short: 2 bytes
long: 8 bytes
long long: 8 bytes

-----

www02$ █
```

Figura 5

35. Inclua, no arquivo ***aula0107.c***, as linhas necessárias para exibir os tamanhos em bytes das combinações válidas entre modificadores de sinal e modificadores de largura, entre tipos básicos e modificadores de sinal, entre tipos básicos e modificadores de largura e entre tipos básicos, modificadores de sinal e modificadores de largura.

Mantenha o formato de exibição do programa anterior, exibindo os títulos correspondentes a cada bloco de informação, ou seja:

```
Tamanhos dos Modificadores de Sinal combinados com Modificadores de Largura

Tamanhos dos Tipos Basicos combinados com Modificadores de Sinal

Tamanhos dos Tipos Basicos combinados com Modificadores de Largura

Tamanhos dos Tipos Basicos combinados com Modificadores de Sinal e com Modificadores de
Largura
```

36. Inclua, no arquivo de dependências, a macro ***AULA0107OBS*** e o rótulo ***aula0107*** com as declarações necessárias. Atualize os valores das macros, onde for necessário.
37. Gere e teste as 20 possíveis versões do executável ***aula0107***.
38. Submeta os arquivos ***aula0107.c*** e os arquivos de dependências ao sistema de controle de versão.
39. Recupere uma cópia de leitura do arquivo ***aula0107.c*** e uma cópia de escrita dos arquivos de dependências.
40. Limpe o diretório (***make clean-all***).

41. Arquivos que devem ser disponíveis ao final da aula:

Subdiretório "~/private/EEL270/2023-2/Aulas-Praticas"

- *aula0101.h*
- *aula0101.c*
- *aula0102.c*
- *aula0103.c*
- *aula0104.c*
- *aula0105.c*
- *aula0106.c*
- *aula0107.c*
- *BSDmakefile*
- *GNUmakefile*

Subdiretório "~/private/EEL270/2023-1/Aulas-Praticas/RCS"

- *aula0101,h,v*
- *aula0101.c,v*
- *aula0102.c,v*
- *aula0103.c,v*
- *aula0104.c,v*
- *aula0105.c,v*
- *aula0106.c,v*
- *aula0107.c,v*
- *BSDmakefile,v*
- *GNUmakefile,v*

Sugestões de Leitura:

1. Página de manual dos comandos *mkdir* e *du*.
2. Página de manual das funções *printf* e *strlen*.
3. Capítulo sobre RCS - livro "Programação para Linux - Aprenda em 24 horas (ver Bibliografia).
4. Capítulo sobre make - livro "Programação para Linux - Aprenda em 24 horas (ver Bibliografia).