

Aula Prática 9

Objetivos:

- Exibição de conteúdo de arquivos, conversão de arquivos do tipo texto entre os formatos DOS e Unix, uso funções *getopt*, *getopt_long* e *getsubopt*.

Versão Inicial: 09/07/2024

Prazo: 16/07/2024 – 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 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**, enquanto que no *Linux* deverá ser definido o argumento **dialecto** 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 dialero=C99** - compila/linkedita usando o **clang** e o dialeto **C99** (somente *Linux*).
 - make cc=GCC dialecto=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 2024 /1
Prof. Marcelo Luiz Drumond Lanza
Autor: <nome completo>

Descricao: <descrição sucinta dos objetivos do programa>
\$Author\$
\$Date\$
\$Log\$

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

\$RCSfile\$

1. Crie o arquivo **aula0901.h** contendo a definição dos tipos **byte** e **tipoErros**. Inclua neste arquivo o protótipo da função **ExibirConteudoArquivo** conforme definido abaixo:

```
tipoErros
ExibirConteudoArquivo (char * /* (E) */);
```

A macro referente à combinação **ifndef** e **define**, como por exemplo **_AULA0901_**, deverá ser definida como uma *string* valendo: **"@(#)aula0901.h \$Revision\$"**

2. Crie o arquivo **aula0901.c** contendo o código-fonte da função **ExibirConteudoArquivo**. Esta função deverá receber um argumento do tipo *string* contendo o nome do arquivo a ser exibido. A função deverá retornar **ok** ou o código de erro apropriado.

Esta função deverá exibir o conteúdo do arquivo, dividindo a tela em três áreas verticais (3 colunas). Na primeira área deverá exibir o *offset* do primeiro byte exibido naquela linha. Os *offsets* deverão ser exibidos em hexadecimal, utilizando sempre 16 dígitos (as letras deverão ser maiúsculas).

A segunda área deverá começar após 1 caractere de espaço, seguido por um caractere pipe (|) e por 1 caractere de espaço. Nesta área deverão ser exibidos os valores em hexadecimal de até 16 bytes (sempre utilizando 2 dígitos e letras maiúsculas). Entre os valores de dois bytes consecutivos, na mesma linha de exibição, deverá ser exibido um e somente um caractere de espaço.

A terceira área deverá começar após 1 caractere de espaço, seguido por um caractere pipe (|) e por 1 caractere de espaço. Na terceira área deverão ser exibidos os caracteres correspondentes aos bytes, se o valor do byte em questão for maior ou igual a 0x20 e menor do que 0x7F. Caso contrário, deverá ser exibido um ponto. Entre 2 caracteres consecutivos não poderá ser exibido nenhum caractere.

A saída gerada por esta função deverá ser semelhante àquela gerada pelo comando **hexdump** com a opção **-C**.

3. Inclua, nos arquivos de dependências, as macros **LIBARQUIVOSOBJS** (correspondendo ao arquivo **aula0901.o**) e **LIBARQUIVOS** (correspondendo ao arquivo **libarquivos.a**). O valor da macro **LIBS** deverá ser atualizado incluindo o valor da macro **LIBARQUIVOS**. Inclua o rótulo correspondente, ou seja, **libarquivos.a**, com a(s) dependência(s) e comando(s) necessários para atingir este objetivo.
4. Crie o arquivo **aula0902.c** contendo o código-fonte de um programa de testes para a função **ExibirConteudoArquivo**. Este programa deverá receber, através dos argumentos da interface de linha de comando, o nome do arquivo desejado. O programa deverá exibir a mensagem de sucesso ou erro correspondente.
5. Inclua, nos arquivos de dependências, as macros **AULA0902OBJS** e **AULA09**. Altere o valor da macro **EXECS**, incluindo o valor da macro **AULA09**. Inclua também os rótulos **aula09** e **aula0902** com os comandos correspondentes (que deverão usar a biblioteca **libarquivos.a**).
6. Gere e teste as 20 versões do executável **aula0902**.
7. Submeta os arquivos **aula0901.h**, **aula0901.c**, **aula0902.c** e ***makefile** ao sistema de controle de versão.

8. Recupere uma cópia de leitura do arquivo **aula0902.c** e uma cópia de escrita dos arquivos **aula0901.h**, **aula0901.c** e ***makefile**.
9. Inclua, no arquivo **aula0901.h**, a definição do protótipo da função *ConverterArquivoFormatoUnixParaFormatoDos*, conforme definido abaixo:

```
tipoErros
ConverterArquivoFormatoUnixParaFormatoDos (char *, /* original */
                                           char * /* convertido */);
```

10. Inclua, no arquivo **aula0901.c**, o código-fonte da função solicitada no item anterior. Esta função deverá receber dois argumentos do tipo *string*. O primeiro argumento deverá conter o nome do arquivo original e o segundo o nome do arquivo que será gerado pela conversão. Se o segundo argumento for igual a **NULL** a função deverá criar um arquivo temporário (usando a função **mkstemp**) e deverá, se executada com sucesso, renomear o arquivo original, incluindo no final a extensão "**backup-AAAAMMDD_hhmmss**" (sem as aspas e substituindo AAAA, MM, DD, hh, mm e ss, pelos valores correspondentes ao ano, mês, dia, hora, minutos e segundos correspondentes ao instante de execução desta função).

A partir do arquivo do tipo texto original (no formato *Unix*) deverá ser gerado o arquivo do tipo texto correspondente (no formato *DOS*). A função deverá retornar **ok** ou o código de erro apropriado.

11. Crie o arquivo **aula0903.c** contendo o código-fonte de um programa de testes para a função criada no item anterior. Este programa deverá receber, através dos argumentos de linha de comando, as informações necessárias para testar a função em questão.
12. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável **aula0903** deverá ser gerado utilizando-se a biblioteca **libarquivos.a**.
13. Gere e teste as 20 versões do executável **aula0903**.
14. Submeta os arquivos **aula0901.h**, **aula0901.c**, **aula0903.c** e ***makefile** ao sistema de controle de versão.
15. Recupere uma cópia de leitura do arquivo **aula0903.c** e uma cópia de escrita dos arquivos **aula0901.h**, **aula0901.c** e ***makefile**.

16. Inclua, no arquivo **aula0901.h**, a definição do protótipo da função *ConverterArquivoFormatoDosParaFormatoUnix*, conforme definido abaixo:

```
tipoErros
ConverterArquivoFormatoDosParaFormatoUnix (char * /* original */ ,
                                           char * /*convertido */);
```

17. Inclua, no arquivo **aula0901.c**, o código-fonte da função solicitada no item anterior. Esta função deverá receber dois argumentos do tipo *string*. O primeiro deverá conter o nome do arquivo original e o segundo o nome do arquivo que será gerado pela conversão. Se o segundo argumento for igual a **NULL** a função deverá criar um arquivo temporário (usando a função **mkstemp**) e deverá, se executada com sucesso, renomear o arquivo original, incluindo no final a extensão "**backup-AAAAMMDD_hhmmss**" (sem as aspas e substituindo AAAA, MM, DD, hh, mm e ss, pelos valores correspondentes ao ano, mês, dia, hora, minutos e segundos correspondentes ao instante de execução desta função).

A partir do arquivo do tipo texto original (no formato *DOS*) deverá ser gerado o arquivo do tipo texto correspondente (no formato *Unix*). A função deverá retornar **ok** ou o código de erro apropriado.

18. Crie o arquivo ***aula0904.c*** contendo o código-fonte de um programa de testes para a função *ConverterArquivoFormatoDosParaFormatoUnix*. Este programa deverá receber, através dos argumentos da interface de linha de comando, as informações necessárias para testar a função em questão.
19. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável ***aula0904*** deverá ser gerado utilizando-se a biblioteca ***libarquivos.a***.
20. Gere e teste as 20 versões do executável ***aula0904***.
21. Submeta os arquivos ***aula0901.h***, ***aula0901.c***, ***aula0904.c*** e ****makefile*** ao sistema de controle de versão.
22. Recupere uma cópia de leitura dos arquivos ***aula0901.h***, ***aula0901.c*** e ***aula0904.c*** e uma cópia de escrita dos arquivos ****makefile***.
23. Crie o arquivo ***aula0905.c*** contendo o código-fonte de um programa de testes para as funções criadas nos itens anteriores. Este programa deverá receber, via argumentos da interface de linha de comando, a opção curta desejada (dentre as permitidas pelo programa). As opções curtas aceitas deverão incluir:
 - d | D - converter um arquivo texto do formato *Unix* para o formato *Microsoft (DOS)*.
 - h | H - exibir uma mensagem contendo as informações sobre o uso do programa.
 - s | S - exibir o conteúdo do arquivo.
 - u | U - converter um arquivo texto do formato *Microsoft* para o formato *Unix*.
24. A implementação deste programa deve utilizar a função ***getopt***.

Considere que todas as opções curtas NÃO possuem argumentos obrigatórios. Nos casos das opções d, D, s, S, u e U, o programa deverá receber via argumentos de linha de comando as demais informações necessárias. Para isso, utilize a variável ***optind***.

Exemplos:

```
./aula0905 -h  
./aula0905 -U aula0901.h  
./aula0905 -d aula0903.c aula0903.c.dos  
./aula0905 -S aula0903
```
25. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável ***aula0905*** deverá ser gerado utilizando-se a biblioteca ***libarquivos.a***.
26. Crie e teste as 20 versões do executável ***aula0905***.
27. Submeta os arquivos ***aula0905.c*** e ****makefile*** ao sistema de controle de versão.

28. Recupere uma cópia de leitura do arquivo **aula0905.c** e uma cópia de escrita dos arquivos ***makefile**.

29. Crie o arquivo **aula0906.c** contendo o código-fonte de um programa de testes para as funções criadas nos itens anteriores. Este programa deverá receber, via argumentos da interface de linha de comando, a opção curta ou longa desejada (dentre as permitidas pelo programa). As opções aceitas deverão incluir:

d | D | dos - converter um arquivo texto do formato *Unix* para o formato *Microsoft (DOS)*.

h | H | help - exibir uma mensagem contendo as informações sobre o uso do programa.

s | S | show - exibir o conteúdo do arquivo.

u | U | unix - converter um arquivo texto do formato *Microsoft* para o formato *Unix*.

A implementação deste programa deve utilizar a função **getopt_long**.

Considere que todas as opções curtas NÃO possuem argumentos obrigatórios. Nos casos das opções d, D, s, S, u e U, o programa deverá receber via argumentos de linha de comando as demais informações necessárias. Para isso, utilize a variável **optind**.

Exemplos:

```
./aula0906 -h
```

```
./aula0906 --dos aula0901.h
```

```
./aula0906 -U aula0903.c aula0903.c.unix
```

```
./aula0906 --show aula0903
```

30. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável **aula0906** deverá ser gerado utilizando-se a biblioteca **libarquivos.a**.

31. Crie e teste as 20 versões do executável **aula0906**.

32. Submeta os arquivos **aula0906.c** e ***makefile** ao sistema de controle de versão.

33. Recupere uma cópia de leitura do arquivo **aula0906.c** e uma cópia de escrita dos arquivos ***makefile**.

34. Crie o arquivo **aula0907.c** contendo o código-fonte de um programa de testes para as funções criadas nos itens anteriores. Este programa deverá receber, via argumentos da interface de linha de comando, a opção curta ou longa desejada (dentre as permitidas pelo programa). As opções aceitas deverão incluir:

d | D | dos - converter um arquivo texto do formato *Unix* para o formato *Microsoft (DOS)*.

h | H | help - exibir uma mensagem contendo as informações sobre o uso do programa.

s | S | show - exibir o conteúdo do arquivo.

u | U | unix - converter um arquivo texto do formato *Microsoft* para o formato *Unix*.

A implementação deste programa deve utilizar as funções **getopt_long** e **getsubopt**.

Considere que todas as opções curtas NÃO possuem argumentos obrigatórios. Nos casos das opções d, D, s, S, u e U, o programa deverá receber via argumentos de linha de comando as demais informações necessárias.

Os argumentos necessários para cada opção deverão ser obtidos usando a função **getsubopt**. As palavras-chave (nomes dos argumentos) utilizadas deverão ser **input** e **output**, correspondendo respectivamente ao nome do arquivo de entrada e ao nome do arquivo de saída.

Exemplos:

```
./aula0907 -h
```

```
./aula0907 --dos input=aula0901.h
```

```
./aula0907 -U input=aula0903.c output=aula0903.c.unix
```

```
./aula0907 --unix output=aula0903.c.unix input=aula0903.c
```

```
./aula0907 --show input=aula0903
```

35. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável **aula0907** deverá ser gerado utilizando-se a biblioteca **libarquivos.a**.
36. Crie e teste as 20 versões do executável **aula0907**.
37. Submeta os arquivos **aula0907.c** e ***makefile** ao sistema de controle de versão.
38. Recupere uma cópia de leitura do arquivo **aula0907.c** e uma cópia de escrita dos arquivos ***makefile**.
39. Limpe o diretório (*make clean-all*)
40. Arquivos que devem ser disponíveis ao final da aula:

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

- **aula0901.h**
- **aula0901.c**
- **aula0902.c**
- **aula0903.c**
- **aula0904.c**
- **aula0905.c**
- **aula0906.c**
- **aula0907.c**
- **BSDmakefile**
- **GNUmakefile**

Além dos arquivos gerados pela ferramenta de controle de versão (localizados no subdiretório **RCS**) e dos arquivos gerados nas aulas práticas anteriores.