Aula Prática 9 - Roteiro

28/09/2021 - Roteiro referente à aula prática 9 - Exibição de conteúdo de arquivos, conversão de arquivos texto entre os formatos DOS e Unix, funções getopt, getopt_long e getsubopt.

Versão: 01/10/2021

Prazo: 11/10/2021 - 18:00

Valor: 10,0 - Peso: 3

- 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 (maiúsculas, minúsculas, caracteres especiais e extensões, se existentes).
- **As tarefas deverão ser executadas na ordem solicitada neste roteiro.**
- Os arquivos de dependências deverão possibilitar que a compilação e 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 o valor padrão para a compilação e para a *linkedição*.

  Para a definição da ferramenta desejada deverá ser definida uma macro (no *FreeBSD*) ou um argumento com o valor desejado (no *CentOS*). As duas macros deverão ser *GCC* e *CLANG* (definidas usando a opção -D). O argumento, identificado por *cc,* deverá ser igual a *GCC* ou *CLANG*.
- Independente da ferramenta utilizada para a compilação, o *flag* de compilação deverá ser definido no instante da execução do comando *make*. O valor padrão para este *flag* deverá ser "-Wall -ansi" (sem as aspas).

  Durante a execução do comando *make* poderão ser definidos outros valores para este *flag* (mantendo a opção de exibir todas as mensagens de advertência) 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, C99 e C11, enquanto que no *CentOS* deverá ser definido o argumento *dialeto* com os valores ANSI, C99 ou C11.
- Crie uma macro, DIALETO, contendo o dialeto a ser utilizado na compilação do código. Esta macro será inicialmente igual a "ansi" e poderá ser alterada para "c99" ou "c11" de acordo com o esquema definido acima.
- O *flag* de linkedição deverá ser igual a "-Wall" (sem as aspas).
- Seguem alguns exemplos:

  make - compila/linkedita (tanto no FreeBSD, quanto no CentOS) com a ferramenta e dialeto padrões, ou seja, gcc e ANSI respectivamente.
  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 CentOS).
  make cc=CLANG - compila/linkedita usando o *clang* e o dialeto ANSI (somente CentOS).
  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 CentOS).
- Inclua, no início de todos os arquivos solicitados, os seguintes comentários:

```
Universidade Federal do Rio de Janeiro
Escola Politecnica
Departamento de Eletronica e de Computacao
EEL270 - Computacao II - Turma 2021/1
```

```
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, os seguintes comentários:

```
$RCSfile$
```

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 colunas. Na primeira coluna deverá exibir o *offset* do primeiro byte exibido naquela linha. Os *offsets* deverão ser exibidos em hexadecimal, utilizando sempre 8 dígitos (as letras deverão ser maiúsculas).

   A segunda coluna deverá começar após 1 caractere de espaço, seguido por um caractere pipe (|) e por 1 caractere de espaço. Nesta coluna 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 coluna deverá começar após 1 caractere de espaço, seguido por um caractere pipe (|) e por 1 caractere de espaço. Na terceira coluna 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.
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 de forma que inclua o valor desta última macro. Inclua o objetivo 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 da *ExibirConteudoArquivo*. Este programa deverá receber, através dos argumentos 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,* de forma que inclua o valor da macro *AULA09*. Inclua também os objetivos *aula09* e *aula0902* com os comandos correspondentes (que deverão usar a biblioteca *libarquivos.a*).
6. Gere e teste as 16 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 *ConverterArquivoFormatoUnixParaFormatoDos*. Esta função deverá receber dois argumentos do tipo *string*. O primeiro contendo 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 texto original (do tipo Unix) deverá ser gerado o arquivo texto correspondente (do tipo 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 *ConverterArquivoFormatoUnixParaFormatoDos*. 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 doze 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 *ConverterArquivoFormatoDosParaFormatoUnix*. Esta função deverá receber dois argumentos do tipo *string*. O primeiro contendo 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 texto original (do tipo DOS) deverá ser gerado o arquivo texto correspondente (do tipo 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 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 16 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 (usando a função *getopt*) 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*.

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 (use a variável *optind*) as demais informações necessárias.

Exemplos:
./aula0905 -h
./aula0905 -U aula0901.h
./aula0905 -d aula0903.c aula0903.c.dos
./aula0905 -S aula0903

24. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável aula0905 deverá ser gerado utilizando-se a biblioteca "*libarquivos.a*".
25. Crie e teste as 16 versões do executável *aula0905*.
26. Submeta os arquivos "*aula0905.c*" e "**makefile"* ao sistema de controle de versão.
27. Recupere uma cópia de leitura do arquivo ""*aula0905.c*" e uma cópia de escrita dos arquivos *"*makefile*".
28. 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 (usando a função *getopt_long*) 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*.

    Considere que todas as opções 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 (use a variável *optind*) as demais informações necessárias.

    Exemplos:
    ./aula0906 -h
    ./aula0906 --dos aula0901.h
    ./aula0906 -U aula0903.c aula0903.c.unix
    ./aula0906 --show aula0903

29. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável aula0906 deverá ser gerado utilizando-se a biblioteca "*libarquivos.a*".
30. Crie e teste as 16 versões do executável *aula0906*.
31. Submeta os arquivos "*aula0906.c*" e "**makefile"* ao sistema de controle de versão.
32. Recupere uma cópia de leitura do arquivo ""*aula0906.c*" e uma cópia de escrita dos arquivos *"*makefile*".
33. 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 (usando a função *getopt_long*) 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*.

    Considere que todas as opções NÃO possuem argumentos obrigatórios. Os argumentos necessários para cada opção deverão ser obtidos usando a função *getsubopt*. As palavras-chave utilizadas deverão ser *input* e *output*, correspondendo respectivamente ao nome do arquivo de entrada e ao nome do arquivo de saída.
    Exemplos:
    ./aula0906 -h
    ./aula0906 --dos input=aula0901.h
    ./aula0906 -U input=aula0903.c output=aula0903.c.unix
    ./aula0906 --unix output=aula0903.c.unix input=aula0903.c

./aula0906 --show input=aula0903

34. Inclua as declarações necessárias nos arquivos de dependências. Lembre-se que o executável aula0907 deverá ser gerado utilizando-se a biblioteca "*libarquivos.a*".
35. Crie e teste as 16 versões do executável *aula0907*.
36. Submeta os arquivos " "*aula0907.c*" e "**makefile"* ao sistema de controle de versão.
37. Recupere uma cópia de leitura do arquivo "*aula0907.c*" e uma cópia de escrita dos arquivos *"*makefile*".
38. Limpe o diretório (*make clean-all*).