

## Aula Prática 4 – Roteiro

### Objetivos:

- Exponencial e Sequência Harmônica Alternada

**Versão Inicial:** 22/05/2023

**Prazo:** 26/05/2023 – 8:00

### 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.**
- 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 **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 dialero=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 2023/1
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 **aula0401.h** contendo o protótipo da função *CalcularExponencial*. Esta função deverá receber uma base (do tipo real) e um expoente (do tipo inteiro) e deverá retornar o valor correspondente à base elevada ao expoente. Lembre-se que o expoente pode ser tanto **negativo**, quanto **positivo**.

Considere que  $0^0$  é igual a 1 e que 0 elevado a números negativos é igual a infinito.

A macro referente à combinação **ifndef** e **define**, por exemplo **AULA0401**, deverá ser definida como uma *string* igual a: "**@(#)aula0401.h \$Revision\$**".

O protótipo da função é definido a seguir:

```
long double
CalcularExponencial (double, int);
```

2. Crie o arquivo **aula0401a.c** contendo a implementação da função *CalcularExponencial*. Esta implementação deverá utilizar **recursividade** e não poderá utilizar nenhuma função de nenhuma biblioteca padrão.

3. Crie o arquivo **aula0402.c** contendo a implementação de um programa de testes para a função *CalcularExponencial*. Este programa deverá receber dois valores através de argumentos da CLI, correspondendo respectivamente à base e ao expoente desejados. Para a conversão de *string* em real utilize a função *strtod* e de *string* em inteiro utilize a função *strtol*. O programa deverá gerar uma saída semelhante à mostrada abaixo:

**base ^ expoente: resultado** (base, espaço, circunflexo, espaço, expoente, dois pontos, espaço)

**2 ^ 2: 4** (fundo azul, texto amarelo, fundo branco, texto vermelho)

4. Inclua, nos arquivos de dependências, as macros **AULA0402AOBJS** e **AULA04**. Altere o valor da macro **EXECS**, de forma que inclua o valor da macro **AULA04**. Inclua também os objetivos **aula04** e **aula0402a** com os comandos correspondentes.
5. Gere e teste as 20 versões do executável **aula0402a**.
6. Submeta os arquivos **aula0401.h**, **aula0401a.c**, **aula0402.c** e **\*makefile** ao sistema de controle de versão.
7. Recupere uma cópia de leitura dos arquivos **aula0401a.c** e **aula0402.c** e uma cópia de escrita do arquivo **aula0401.h** e dos arquivos de dependência.
8. Adicione **aula0401a.o** ao valor da macro **LIBMATEMATICARECURSAO OBJS**.
9. Gere o arquivo **libmatematicarecursao.a**.
10. Crie o arquivo **aula0401b.c** contendo a implementação da função *CalcularExponencial*. Esta implementação deverá utilizar o laço de repetição **do ... while** e não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
11. Inclua, nos arquivos de dependências, a macro **AULA0402BOBJS** e o objetivo **aula0402b** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
12. Gere e teste as 20 versões do executável **aula0402b**.
13. Submeta os arquivos **aula0401b.c** e **\*makefile** ao sistema de controle de versão.
14. Recupere uma cópia de leitura do arquivo **aula0401b.c** e uma cópia de escrita dos arquivos de dependências.
15. Adicione **aula0401b.o** ao valor da macro **LIBMATEMATICADOWHILE OBJS**.
16. Gere o arquivo **libmatematicadowhile.a**.
17. Crie o arquivo **aula0401c.c** contendo a implementação da função *CalcularExponencial*. Esta implementação deverá utilizar o laço de repetição **for** e não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
18. Inclua, nos arquivos de dependências, a macro **AULA0402COBJS** e o objetivo **aula0402c** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
19. Gere e teste as 20 versões do executável **aula0402c**.
20. Submeta os arquivos **aula0401c.c** e **\*makefile** ao sistema de controle de versão.

21. Recupere uma cópia de leitura do arquivo **aula0401c.c** e uma cópia de escrita dos arquivos de dependências.
22. Adicione **aula0401c.o** ao valor da macro **LIBMATEMATICAFOROBJJS**.
23. Gere o arquivo **libmatematicafor.a**.
24. Crie o arquivo **aula0401d.c** contendo a implementação da função *CalcularExponencial*. Esta implementação deverá utilizar o laço de repetição **while** e não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
25. Inclua, nos arquivos de dependências, a macro **AULA0402DOBJJS** e o objetivo **aula0402d** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
26. Gere e teste as 20 versões do executável **aula0402d**.
27. Submeta os arquivos **aula0401d.c** e **\*makefile** ao sistema de controle de versão.
28. Recupere uma cópia de leitura do arquivo **aula0401d.c** e uma cópia de escrita dos arquivos de dependências.
29. Adicione **aula0401d.o** ao valor da macro **LIBMATEMATICAWHILEOBJJS**.
30. Gere o arquivo **libmatematicawhile.a**.
31. Inclua, no arquivo **aula0401.h**, o protótipo da função *CalcularSerieHarmonicaAlternada*. Esta função deverá receber um número inteiro não negativo representando o número de termos que deverá ser utilizado para calcular o valor da série harmônica alternada. A função deverá retornar o valor calculado. Considere que:
 
$$S(n) = 0 \text{ se } n \text{ é igual a } 0.$$

$$S(n) = 1 - 1/2^2 + 1/3^3 - 1/4^4 + \dots - 1/n^n \text{ se } n \text{ é par}$$

$$S(n) = 1 - 1/2^2 + 1/3^3 - 1/4^4 + \dots + 1/n^n \text{ se } n \text{ é ímpar}$$
 O protótipo da função é definido a seguir:
 

```
float
CalcularSerieHarmonicaAlternada (unsigned short int);
```
32. Crie o arquivo **aula0403a.c** contendo a implementação da função definida no item anterior. A implementação desta função deverá utilizar **recursividade** e a função *CalcularExponencial* (através da biblioteca **libmatematicarecursao.a**). Esta implementação não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
33. Crie o arquivo **aula0404.c** contendo a implementação de um programa de testes para a função *CalcularSerieHarmonicaAlternada*. Este programa deverá receber, através de argumentos da CLI, um valor **maior do que 0** e **menor do que 1** representando um percentual **P**.
 

O programa deverá exibir o valor da série (sempre com 10 casas decimais) variando n de 0 até N, enquanto  $|S(N) - S(N-1)|$  for maior ou igual a  $P * S(N - 1)$ .

$S(N) - S(N - 1)$  pode ser positivo ou negativo.  $|S(N) - S(N-1)|$  representa o valor absoluto desta diferença. Para obter o valor absoluto utilize a função **fabsf**.

Para a conversão do argumento de linha de comando em real utilize a função *strtod*.

O programa deverá gerar uma saída semelhante à mostrada abaixo (valor para n ímpar com a cor do fundo invertida):

./aula0402a 0.30

S(0): 0.000000000000

S(1): 1.000000000000

S(2): 0.750000000000

34. Inclua, nos arquivos de dependências, a macro *AULA0404AOBJS* e o objetivo *aula0404a* com os comandos correspondentes. Altere o valor da macro *AULA04*, incluindo o binário correspondente. Para usar uma biblioteca estática inclua, no final da linha de *linkedição*, o seguinte: *-L. -INOMEDABIBLIOTECA*, onde *NOMEDABIBLIOTECA* corresponde ao nome do arquivo correspondente sem o prefixo *lib* e sem o sufixo *.a*.
35. Gere e teste as 20 versões do executável *aula0404a* utilizando a biblioteca criada.
36. Submeta os arquivos *aula0401.h*, *aula0403a.c*, *aula0404.c* e *\*makefile* ao sistema de controle de versão.
37. Recupere uma cópia de leitura dos arquivos *aula0401.h*, *aula0403a.c* e *aula0404.c* e uma cópia de escrita dos arquivos e dos arquivos de dependências.
38. Adicione *aula0403a.o* ao valor da macro *LIBMATEMATICARECURSAOOBJS*.
39. Gere o arquivo *libmatematicarecursao.a*.
40. Crie o arquivo *aula0403b.c* contendo a implementação da função definida no item 31. A implementação desta função deverá utilizar o laço de repetição *do ... while* e a função *CalcularExponencial* (através da biblioteca *libmatematicadownhile.a*). Esta implementação não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
41. Inclua, nos arquivos de dependências, a macro *AULA0404BOBJS* e o objetivo *aula0404b* com os comandos correspondentes. Altere o valor da macro *AULA04*, incluindo o binário correspondente.
42. Gere e teste as 20 versões do executável *aula0404b* utilizando a biblioteca criada.
43. Submeta os arquivos *aula0403b.c* e *\*makefile* ao sistema de controle de versão.
44. Recupere uma cópia de leitura do arquivo *aula0403b.c* e uma cópia de escrita dos arquivos *\*makefile*.
45. Adicione *aula0403b.o* ao valor da macro *LIBMATEMATICADOWHILEOBJS*.
46. Gere o arquivo *libmatematicadownhile.a*.
47. Crie o arquivo *aula0403c.c* contendo a implementação da função definida no item 31. A implementação desta função deverá utilizar o laço de repetição *for* e a função *CalcularExponencial* (através da biblioteca *libmatematicafor.a*). Esta implementação não poderá utilizar nenhuma função de nenhuma biblioteca padrão.

48. Inclua, nos arquivos de dependências, a macro **AULA0404COBJS** e o objetivo **aula0404c** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
49. Gere e teste as 20 versões do executável **aula0404c** utilizando a biblioteca criada.
50. Submeta os arquivos **aula0403c.c** e **\*makefile** ao sistema de controle de versão.
51. Recupere uma cópia de leitura do arquivo **aula0403c.c** e uma cópia de escrita dos arquivos de dependências.
52. Adicione **aula0403c.o** ao valor da macro **LIBMATEMATICAFOROBJJS**.
53. Gere o arquivo **libmatematicafor.a**.
54. Crie o arquivo **aula0403d.c** contendo a implementação da função definida no item 31. Esta implementação deverá utilizar o laço de repetição **while** e a função **CalcularExponencial** (através da biblioteca **libmatematicawhile.a**). Esta implementação não poderá utilizar nenhuma função de nenhuma biblioteca padrão.
55. Inclua, nos arquivos de dependências, a macro **AULA0404DOBJJS** e o objetivo **aula0404d** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
56. Gere e teste as 20 versões do executável **aula0404d** utilizando a biblioteca criada.
57. Submeta os arquivos **aula0403d.c** e **\*makefile** ao sistema de controle de versão.
58. Recupere uma cópia de leitura do arquivo **aula0403d.c** e uma cópia de escrita dos arquivos de dependências.
59. Adicione **aula0403d.o** ao valor da macro **LIBMATEMATICAWHILEOBJJS**.
60. Gere o arquivo **libmatematicawhile.a**.
61. Crie o arquivo **aula0403e.c** contendo a implementação da função definida no item 31. Esta implementação deverá utilizar recursividade ou o laço de repetição desejado e a função da biblioteca **math** mais adequada para o cálculo da exponencial necessária. Leia com atenção a página de manual da função e verifique como fazer a **linkedição** com a função escolhida.
62. Inclua, nos arquivos de dependências, a macro **AULA0404EOBJJS** e o objetivo **aula0404e** com os comandos correspondentes. Altere o valor da macro **AULA04**, incluindo o binário correspondente.
63. Gere e teste as 20 versões do executável **aula0404e**.
64. Submeta os arquivos **aula0403e.c** e **\*makefile** ao sistema de controle de versão.
65. Recupere uma cópia de leitura do arquivo **aula0403e.c"** e uma cópia de escrita dos arquivos de dependências.
66. Limpe o diretório (**make clean-all**).
67. Arquivos que devem ser disponíveis ao final da aula:  
Subdiretório **"~/private/EEL270/2023-1/Aulas-Praticas"**

- *aula0401.h*
- *aula0401a.c*
- *aula0401b.c*
- *aula0401c.c*
- *aula0401d.c*
- *aula0402.c*
- *aula0403a.c*
- *aula0403b.c*
- *aula0403c.c*
- *aula0404d.c*
- *aula0403e.c*
- *aula0404.c*
- *BSDmakefile*
- *GNUmakefile*

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