Aula Prática 06 - Roteiro

30/04/2019 - Roteiro referente à aula prática 06 - Geração e Validação de Números PIS/PASEP.

Versão: 30/04/2019

Prazo: 07/05/2019 - 8:00

Observações:

- Leia este enunciado com **MUITA** atenção até o final antes de iniciar o trabalho.
- Este roteiro está disponível no formato PDF. Para acessá-lo, clique aqui.
- 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.
- A compilação e a *linkedição* deverão ser executadas utilizando-se tanto o *gcc*, quanto o *clang* . Em ambos os casos deverão ser utilizados os flags "*-Wall -std=c99*".
- Além disso, deverão ser executadas sem mensagens de advertência e sem mensagens de erro, tanto no *CentOS* 7.x, quanto no *FreeBSD* 11.x.
- No *CentOS* o comando *make* corresponde ao *GNU Make*, enquanto que no *FreeBSD* o comando é nativo. Estas duas variantes não são cem por cento compatíveis e por isso serão necessários dois arquivos de dependências, o *GNUmakefile* e o *BSDmakefile*. No *FreeBSD* o comando *gmake* poderia ser utilizado com o arquivo *GNUmakefile*, mas isto está fora do escopo desta aula.
- Inclua, sempre que necessário, o comando para criar uma cópia do binário com a identificação do sistema operacional e do *compilador/linkeditor* utilizados.

- Inclua, no início de todos os arquivos solicitados (*.c e *makefile), os seguintes comentários:

```
Universidade Federal do Rio de Janeiro
Escola Politecnica
Departamento de Eletronica e de Computacao
EEL270 - Computacao II - Turma 2019/1
Prof. Marcelo Luiz Drumond Lanza
Autor: <nome completo>
Descricao: <descrição sucinta dos objetivos do programa>

$Author$
$Date$
$Log$
```

O PIS/PASEP é um identificador composto por dez dígitos mais um dígito verificador. Este dígito verificador é calculado através da seguinte regra:

- Multiplique os números, da esquerda para a direita, respectivamente por 3, 2, 9, 8, 7, 6, 5, 4, 3 e 2.
- Some os resultados das multiplicações.
- Calcule o resto da divisão desta soma por 11.
- Se o resto for igual a 0 ou igual a 1 o dígito verificador será igual a 0.
- Caso contrário, o dígito verificador será igual à diferença entre 11 e o resto calculado acima.

Como exemplo, para o número 1701209041-1 , o cálculo seria:

1 x 3 + 7 x 2 + 0 x 9 + 1 x 8 + 2 x 7 + 0 x 6 + 9 x 5 + 0 x 4 + 4 x 3 + 1 x 2 = 98

Como o resto da divisão de 98 por 11 é 10 e como a diferença entre 11 e este resto é 1, o dígito verificador será igual

ao resto, ou seja, 1.

1. Baseado nas definições acima, crie o arquivo "*aula0601.h*" contendo o protótipo da função GerarDigitoVerificadorPisPasep. Este arquivo deverá conter também as macros e os tipos necessários para a implementação desta função. A macro referente à combinação *ifndef* e *define*, como por exemplo _AULA0601_, deverá ser definida como uma *string* valendo:

   "@(#)aula0601.h $Revision$"

   tipoErros
   GerarDigitoVerificadorPisPasep (byte [ ]);

   A função *GerarDígitoVerificadorPisPasep* deverá receber os 10 primeiros dígitos de um número PIS/PASEP (como um vetor de *bytes*) e deverá devolver o dígito verificador correspondente (na décima primeira posição do vetor recebido).

2. Crie o arquivo "*aula0601.c*" contendo a implementação da função *GerarDigitoVerificadorPisPasep*.

3. Crie o arquivo "*aula0602a.c*" contendo a implementação de um programa de testes para a função *GerarDigitoVerificadorPisPasep*. Este programa deverá receber os 10 primeiros dígitos do número PIS/PASEP desejado através de 10 argumentos da linha de comando (CLI). O programa deverá exibir o dígito verificador correspondente. Todos os tratamentos de erro necessários e que não possam realizados na função *GerarDigitoVerificadorPisPasep* deverão ser implementados neste programa.

4. Inclua, nos arquivos de dependências, a macro AULA0602AOBJS - correspondendo aos arquivos necessários para criar o executável a partir dos arquivos "*aula0601.c*" e "*aula0602a.c*". Além disso, defina a macro AULA06, equivalendo ao executável *"aula0602a*", e altere a macro EXECS, de forma que o valor da mesma inclua os executáveis criados na aula 6. O arquivo de dependências deverá incluir ainda os objetivos *aula06* (executável da aula 6) e *aula0602a* (executável criado a partir dos arquivos definidos pela macro AULA0602AOBJS) com os comandos correspondentes.
5. Crie e teste as quatro versões do executável *aula0602a.*
6. Submeta os arquivos "*aula0601.h*", *"aula0601.c"*, "*aula0602a.c*" e "**makefile"* ao sistema de controle de versão.
7. Recupere uma cópia de leitura do arquivo "*aula0602a.c"* e uma cópia de escrita dos arquivos *"aula0601.h*", "*aula0601.c"* e *"*makefile*".
8. Crie o arquivo "*aula0602b.c*" contendo a implementação de um programa de testes para a função *GerarDigitoVerificadorPisPasep*. Este programa deverá receber os 10 primeiros dígitos do número PIS/PASEP desejado através de um único argumento da linha de comando (CLI). O programa deverá exibir o dígito verificador correspondente. Todos os tratamentos de erro necessários e que não possam realizados na função *GerarDigitoVerificadorPisPasep* deverão ser implementados neste programa.
9. Inclua, nos arquivos de dependências, a macro AULA0602BOBJS - correspondendo aos arquivos necessários para criar o executável a partir dos arquivos "*aula0601.c*" e "*aula0602b.c*". Além disso, altere a macro AULA06, de forma que o valor da mesma inclua o executável "*aula0602b*". O arquivo de dependências deverá incluir ainda o objetivo *aula0602b* (executável criado a partir dos arquivos definidos pela macro AULA0602BOBJS) com os comandos correspondentes.
10. Crie e teste as quatro versões do executável *aula0602b.*
11. Submeta os arquivos "*aula0601.h*", *"aula0601.c"*, "*aula0602b.c*" e "**makefile"* ao sistema de controle de versão.
12. Recupere uma cópia de leitura dos arquivos *"aula0601.h*", "*aula0601.c"* e "*aula0602b.c"* e uma cópia de escrita do arquivo *"*makefile*".
13. Inclua, no arquivo "*aula0601.h*", o protótipo da função ValidarPisPasep e a definição dos tipos necessários.

    tipoErros
    ValidarPisPasep (byte [ ]);

    A função *ValidarPisPasep* deverá receber os 11 dígitos de um número PIS/PASEP (como um vetor de bytes) e deverá retornar *ok* se o dígito verificador for válido ou o código de erro correspondente Esta função deverá utilizar a função *GerarDigitoVerificadorPisPasep*.

14. Inclua, no arquivo "*aula0601.c*", a implementação da função *ValidarPisPasep*.

15. Crie o arquivo "*aula0603a.c*" contendo a implementação de um programa de testes para a função *ValidarPisPasep*.Este programa deverá receber os 11 dígitos do PIS/PASEP desejado através de 11 argumentos da linha de comando (CLI) e deverá exibir se o PIS/PASEP em questão é válido ou inválido. Todos os tratamentos de erro necessários e que não possam realizados na função *ValidarPisPasep* deverão ser implementados neste programa.
16. Inclua, nos arquivos de dependências, a macro AULA0603AOBJS - correspondendo aos arquivos necessários para criar o executável a partir dos arquivos "*aula0601.c*" e "*aula0603a.c*". Além disso, altere a macro AULA06, de forma que o valor da mesma inclua o executável "*aula0603a*". O arquivo de dependências deverá incluir ainda o objetivo *aula0603a* (executável criado a partir dos arquivos definidos pela macro AULA0603AOBJS) com os comandos correspondentes.
17. Crie e teste as quatro versões do executável *aula0603a.*
18. Submeta os arquivos "*aula0601.h*", *"aula0601.c"*, "*aula0603a.c*" e "**makefile"* ao sistema de controle de versão.
19. Recupere uma cópia de leitura dos arquivos *"aula0601.h*", "*aula0601.c"* e "*aula0603a.c"* e uma cópia de escrita do arquivo *"*makefile*".
20. Crie o arquivo "*aula0603b.c*" contendo a implementação de um programa de testes para a função *ValidarPisPasep*. Este programa deverá receber os 11 dígitos do PIS/PASEP desejado através de um único argumento da linha de comando (CLI) e deverá exibir se o PIS/PASEP em questão é válido ou inválido. O argumento recebido deverá incluir o hífen. Todos os tratamentos de erro necessários e que não possam realizados na função *ValidarPisPasep* deverão ser implementados neste programa.
21. Inclua, nos arquivos de dependências, a macro AULA0603BOBJS - correspondendo aos arquivos necessários para criar o executável a partir dos arquivos "*aula0601.c*" e "*aula0603b.c*". Além disso, altere a macro AULA06, de forma que o valor da mesma inclua o executável "*aula0603b*". O arquivo de dependências deverá incluir ainda o objetivo *aula0603b* (executável criado a partir dos arquivos definidos pela macro AULA0603BOBJS) com os comandos correspondentes.
22. Crie e teste as quatro versões do executável *aula0602b.*
23. Submeta os arquivos "*aula0601.h*", *"aula0601.c"*, "*aula0603b.c*" e "**makefile"* ao sistema de controle de versão.
24. Recupere uma cópia de leitura dos arquivos *"aula0601.h*", "*aula0601.c"* e "*aula0603b.c"* e uma cópia de escrita do arquivo *"*makefile*".
25. Repita todos os ítens anteriores trocando o vetor de bytes por uma *string*:

    tipoErros
    *GerarDigitoVerificadorPisPasep (char * /* entrada */ , char * /* saida */);*

    tipoErros
    *ValidarPisPasep (char *);*

    A função *ValidarPisPasep* deverá receber uma *string* que inclua o hífen.

    Não se esqueça de renumerar corretamente os arquivos, ou seja:
    - "*aula0601.h*" será renomeado para "*aula0604.h*".
    - "*aula0601.c*" será renomeado para "*aula0604.c*".
    - "*aula0602a.c*" será renomeado para "*aula0605a.c*".
    - "*aula0602b.c*" será renomeado para "*aula0605b.c*".
    - "*aula0603a.c*" será renomeado para "*aula0606a.c*".
    - "*aula0603b.c*" será renomeado para "*aula0606b.c*".

Não se esqueça de incluir as macros e rótulos necessários nos arquivos de dependências.