UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL
IMD0030 – LINGUAGEM DE PROGRAMAÇÃO I

Laboratório 4
Manipulação de arquivos

Objetivo

O objetivo deste exercício é colocar em prática as operações de leitura e escrita em arquivos na linguagem de programação C++. O programa a ser implementado utiliza como fonte de dados um arquivo CSV (*Comma-Separated Values*)[1] contendo o número de nascidos vivos em cada um dos 167 municípios do Estado do Rio Grande do Norte entre os anos de 1994 e 2014. Os dados são provenientes do Sistema de Informações sobre Nascidos Vivos (SINASC) do Departamento de Informática do Sistema Único de Saúde (DATASUS), vinculado ao Ministério da Saúde do Brasil, e estão disponíveis em http://tabnet.datasus.gov.br/cgi/tabcgi.exe?sinasc/cnv/nvuf.def.

Orientações gerais

Você deverá observar as seguintes observações gerais na implementação deste exercício:

1) Apesar da completa compatibilidade entre as linguagens de programação C e C++, seu código fonte **não** deverá conter recursos da linguagem C nem ser resultante de mescla entre as duas linguagens, o que é uma má prática de programação. Dessa forma, deverão ser utilizados **estritamente** recursos da linguagem C++.

2) Durante a compilação do seu código fonte, você deverá habilitar a exibição de mensagens de aviso (*warnings*), pois elas podem dar indícios de que o programa potencialmente possui problemas em sua implementação que podem se manifestar durante a sua execução.

3) Aplique boas práticas de programação. Codifique o programa de maneira legível (com indentação de código fonte, nomes consistentes, etc.) e documente-o adequadamente na forma de comentários. Anote ainda o código fonte para dar suporte à geração automática de documentação utilizando a ferramenta Doxygen (http://www.doxygen.org/). Consulte o documento extra disponibilizado na Turma Virtual do SIGAA com algumas instruções acerca do padrão de documentação e uso do Doxygen.

4) Busque desenvolver o seu programa com qualidade, garantindo que ele funcione de forma correta e eficiente. Pense também nas possíveis entradas que poderão ser utilizadas para testar apropriadamente o seu programa e trate adequadamente possíveis entradas consideradas inválidas.

[1] *Comma-separated values* (Wikipedia): https://en.wikipedia.org/wiki/Comma-separated_values

5) Lembre-se de aplicar boas práticas de modularização, em termos da implementação de diferentes funções e separação entre arquivos cabeçalho (`.h`) e corpo (`.cpp`).
6) A fim de auxiliar a compilação do seu projeto, construa **obrigatoriamente** um `Makefile` que faça uso da estrutura de diretórios apresentada anteriormente em aula.
7) A organização deste exercício na forma de tarefas tem finalidade apenas didática, de modo que **apenas um programa executável** deverá ser gerado e entregue após a conclusão das tarefas.
8) Garanta o uso consistente de alocação dinâmica de memória. Para auxilia-lo nesta tarefa, você pode utilizar o Valgrind (http://valgrind.org/) para verificar se existem problemas de gerenciamento de memória.

Autoria e política de colaboração

Este trabalho deverá ser realizado **individualmente**. O trabalho em cooperação entre estudantes da turma é estimulado, sendo admissível a discussão de ideias e estratégias. Contudo, tal interação não deve ser entendida como permissão para utilização de (parte de) código fonte de colegas, o que pode caracterizar situação de plágio. Trabalhos copiados em todo ou em parte de outros colegas ou da Internet serão sumariamente rejeitados e receberão nota zero.

Apesar de o trabalho ser feito individualmente, você deverá utilizar o sistema de controle de versões Git no desenvolvimento. Ao final, todos os arquivos de código fonte do repositório Git local deverão estar unificados em um repositório remoto Git hospedado em algum serviço da Internet, a exemplo do GitHub, Bitbucket, Gitlab ou outro de sua preferência. A fim de garantir a boa manutenção de seu repositório, configure corretamente o arquivo `.gitignore` em seu repositório Git.

Entrega

Você deverá submeter um único arquivo compactado no formato `.zip` contendo todos os códigos fonte resultantes da implementação deste exercício, sem erros de compilação e devidamente testados e documentados, **até as 12h do dia 13 de abril de 2017** através da opção *Tarefas* na Turma Virtual do SIGAA. Você deverá ainda informar, no campo *Comentários* do formulário de submissão da tarefa, o endereço do repositório Git utilizado.

Avaliação

O trabalho será avaliado sob os seguintes critérios: (i) utilização correta dos conteúdos vistos anteriormente e nas aulas presenciais da disciplina; (ii) a corretude da execução dos programas implementados, que devem apresentar saída em conformidade com a especificação e as entradas de dados fornecidas; (iii) a aplicação correta de boas práticas de programação, incluindo legibilidade, organização e documentação de código fonte, e; (iv) a utilização correta do repositório Git, no qual deverá estar registrado todo o histórico da implementação por meio de *commits*. A presença de mensagens de aviso (*warnings*) ou de erros de compilação e/ou de execução, a modularização inapropriada e a ausência de documentação são faltas que serão penalizadas. Este trabalho contabilizará nota de até 1,0 ponto na 1ª Unidade da disciplina.

Tarefa 1

Implemente um programa chamado `nascimentos` que recebe como entrada, via linha de comando, um arquivo de texto no formato CSV (disponível para *download* através da Turma Virtual do SIGAA) contendo os números de nascidos vivos em cada município para cada ano contabilizado. Cada linha do arquivo refere-se a um município e os números de nascimentos em cada ano são separados por ponto-e-vírgulas. Enquanto o arquivo de entrada for sendo lido, os dados de cada município deverão ser armazenados em um vetor alocado dinamicamente em memória com objetos do tipo estrutura `Stats`, que agrega os dados referentes aos nascimentos em um município:

```
/**
 * @struct   Stats stats.h
 * @brief    Tipo estrutura que agrega os dados de nascimento de um município
 * @details  Os dados cobrem os anos de 1994 a 2014
 */
struct Stats {
    string codigo;         /**< Codigo do municipio */
    string nome;           /**< Nome do municipio */
    int nascimentos[21];   /**< Numero de nascimentos em cada ano contabilizado */
};
```

Observações importantes:

1) Ao abrir o arquivo CSV de entrada em um editor de texto ou editor de planilhas, será possível observar que a primeira linha do arquivo diz respeito a um cabeçalho para a tabela de dados e que a primeira coluna contém o nome do município antecedido por um código numérico que o identifica. Durante a leitura do arquivo, o programa deverá separar o código numérico do nome do município, cada um sendo armazenado no respectivo campo do tipo estrutura `Stats`.
2) O programa deverá desprezar a última coluna durante o carregamento dos dados, de modo que os objetos do tipo estrutura `Stats` armazenarão **apenas** o código e o nome do município e os números de nascidos nesse município em cada ano.
3) A última linha e a última coluna do arquivo CSV de entrada apresentam, respectivamente, o total de nascimentos em cada ano e o total de nascimentos em cada município somados os 21 anos da série 1994-2014. No entanto, esses totais deverão ser usados **apenas** para fins de verificação de consistência. Para isso, você deverá implementar uma função de verificação para verificar se os totais informados no arquivo conferem com a série de dados lida.

Após a leitura do arquivo, o programa deverá computar as seguintes estatísticas que resumem o conjunto de dados em questão:

(i) o *maior* número de nascimentos em cada ano;
(ii) o *menor* número de nascimentos em cada ano;
(iii) a *média* do número de nascimentos em cada ano;
(iv) o *desvio padrão*[2] do número de nascimentos em cada ano, e;

[2] Desvio padrão (Wikipedia): https://pt.wikipedia.org/wiki/Desvio_padrão

(v) o número *total* de nascimentos em cada ano.

Para calcular o desvio padrão σ do número de nascimentos em um determinado ano, você poderá utilizar a seguinte equação:

$$\sigma = \sqrt{\frac{1}{M} \times \sum_{i=1}^{M} (n_i - MD)^2}$$

em que M é o número de municípios, n_i é o número de nascimentos do i-ésimo município no ano em questão e MD é a média do número de nascimentos nesse ano. Um baixo desvio padrão indica que os pontos dos dados tendem a estar próximos da média do, enquanto que um alto desvio padrão indica que os pontos dos dados estão espalhados por uma ampla gama de valores.

Como saída, o programa deverá gerar automaticamente dois arquivos:

(i) um arquivo de texto no formato CSV chamado `estatisticas.csv`, no qual cada linha corresponde a um ano e suas respectivas estatísticas acerca do número de nascimentos, cada valor sendo separado por ponto-e-vírgulas, e;
(ii) um arquivo de texto chamado `totais.dat` contendo **apenas** o ano e o respectivo número total de nascimentos nesse ano, separados por espaço.

Uma vez gerado o arquivo `totais.dat` como saída da execução do programa implementado, você deverá utilizar o programa `gnuplot` (http://www.gnuplot.info/) para gerar automaticamente um histograma[3] que mostrará a evolução do número de nascimentos entre os anos de 1994 e 2014. Após a instalação do `gnuplot`, o histograma pode ser gerado executando-se o seguinte comando no Terminal do sistema operacional Linux:

```
$ gnuplot -e "filename='totais.dat'" histograma.gnuplot
```

Onde, `totais.dat` é o arquivo anteriormente gerado e que contém os dados a serem plotados no histograma, enquanto que o arquivo `histograma.gnuplot` é um *script* de configuração para instrução do `gnuplot` quando da geração do gráfico. Esse *script* deverá ter o seguinte conteúdo:

[3] Histograma (Wikipedia): https://pt.wikipedia.org/wiki/Histograma

```
# Inicializacao
clear
reset
set key off

# Configuracoes de saida: inclui formato de exportacao, tamanho do grafico,
# fontes utilizadas e nome do arquivo de saida

# Exportacao para o formato .png
set terminal png size 640,480 enhanced font 'Helvetica,12'
set output 'histograma.png'

# Exportacao para o formato .jpg
# set terminal jpeg size 640,480 enhanced font 'Helvetica' 12
# set output 'histograma.jpg'


# Exportação para o formato .svg
# set terminal svg size 640,480 enhanced background rgb 'white' fname 'Helvetica' fsize
14 butt solid
# set output 'histograma.svg'

# Título do gráfico
set title 'Total de nascidos vivos no RN (1994-2014)'

# Configurações do eixo horizontal
set xrange[1994:2014]                   # Faixa de valores
set xtics 1                             # Salto entre valores
set xtic rotate by -45 scale 0          # Rotação dos rótulos

# Configurações do eixo vertical
set yrange[0:80000]                     # Faixa de valores

# Seleção do tipo de gráfico a ser gerado (histograma)
set style data histogram
set style histogram clustered gap 1
set style fill solid border -1          # Preenchimento e contorno
set linetype 1 lc rgb 'green'           # Cor
set boxwidth 0.6                        # Largura das barras verticais

# Plotagem do gráfico
# Os dados a serem plotados constam no arquivo totais.dat
plot 'totais.dat' using 1:2 title '' smooth freq with boxes
```

Por padrão, esse *script* gerará um gráfico na forma de um arquivo de imagem no formato PNG, a saber, `histograma.png`. Para gerar em outros formatos de arquivos de imagem, você deverá remover o caractere # (que indica comentário) das respectivas linhas responsáveis pela geração no formato em questão, iniciadas pelos comandos `set terminal` e `set output`. É importante destacar que **não é possível** gerar mais de um arquivo de imagem em uma única plotagem (comando `plot`), ou seja, é necessário repetir o comando de plotagem de dados para cada arquivo de imagem a ser gerado, caso deseje-se gerar saída em múltiplos formatos.

Tarefa 2

Modifique o programa implementado na Tarefa 1 para que ele seja capaz de fornecer, na saída padrão, respostas às seguintes perguntas:

a) Qual município apresentou a maior *taxa de queda* no número de nascimentos quando comparados os anos de 2013 e 2014?
b) Qual município apresentou a maior *taxa de crescimento* no número de nascimentos quando comparados os anos de 2013 e 2014?

A taxa de crescimento relativa TC de cada município pode ser calculada através da seguinte equação:

$$TC = \frac{N_{2014}}{N_{2013}}$$

em que N_{2014} e N_{2013} são respectivamente os números de nascidos vivos nos anos de 2014 e 2013. Caso o valor de TC seja inferior a 1, tem-se então uma taxa de queda (ou taxa de crescimento negativa).

Um exemplo de execução dessa nova versão do programa seria:

```
$ ./estatisticas Nascimentos_RN.csv
... Arquivo estatisticas.csv gerado
... Arquivo totais.dat gerado

Municipio com maior taxa de queda 2013-2014: Pedro Avelino (-26.32%)
Municipio com maior taxa de crescimento 2013-2014: Macau (+160.8%)
```

As porcentagens de crescimento (ou queda) P apresentadas na saída do programa, que deverão considerar estritamente duas casas decimais, podem ser calculadas através da seguinte equação:

$$P = 100 \times (TC - 1)$$

Tarefa EXTRA (+50%)

Um cliente muito especial ($$$$) contratou você para filtrar dados do mesmo conjunto de dados que você usou neste Laboratório. O seu cliente tem especial interesse nos municípios da Microrregião do Seridó Oriental do Rio Grande do Norte. Lá ele deseja instalar um centro comercial que demandará muita mão-de-obra local e clientes. Por essa razão, ele quer estudar o comportamento das populações desses municípios a fim de decidir o local mais indicado para construir o seu empreendimento. Após algumas reuniões, ficou decidido que você irá apresentar a progressão dessas populações através de um **gráfico de linha**, mostrando a *taxa de crescimento* para cada um dos municípios em toda a série histórica, ou seja, para cada ano amostrado.

Nas reuniões que você teve com o seu cliente, você notou que ele não parecia decidido sobre a Microrregião do Seridó Oriental. Já pensando na possibilidade de ele requisitar uma nova pesquisa em outra região, você decidiu que o seu programa será configurável através de um arquivo chamado `alvos.dat`, que conterá apenas uma lista com os códigos dos municípios (um por linha) a serem considerados para o gráfico de linhas. Um exemplo de conteúdo para o arquivo `alvos.dat` considerando alguns municípios da Região Metropolitana de Natal seria:

```
240810
240325
240360
241200
240260
```

em que esses códigos se referem respectivamente aos municípios de Natal, Parnamirim, Extremoz, São Gonçalo do Amarante e Ceará-Mirim.

Um exemplo de execução desse programa seria:

```
$ ./extra Nascimentos_RN.csv
... Arquivo estatisticas.csv gerado
... Lendo arquivo alvos.dat

......[5] municípios definidos como alvo
......{ Natal }
......{ Parnamirim }
......{ Extremoz }
......{ São Gonçalo do Amarante }
......{ Ceará-Mirim }
... Arquivo extra.dat gerado
```

A geração automática do gráfico de linha fazendo uso do `gnuplot` deverá seguir a mesma ideia desenvolvida na Tarefa 1 em termos de utilizar um arquivo de entrada (no caso o arquivo `extra.dat`) contendo os dados a serem plotados no gráfico. Para possibilitar a geração do gráfico pelo `gnuplot`, você deverá criar um *script* de configuração próprio, podendo inclusive adaptar o arquivo `histograma.gnuplot` apresentado anteriormente. Além da documentação oficial do `gnuplot` disponível em http://gnuplot.info/documentation.html, bons exemplos de uso e demonstração podem ser encontrados nos endereços http://alvinalexander.com/technology/gnuplot-charts-graphs-examples e http://gnuplot.sourceforge.net/demo/.