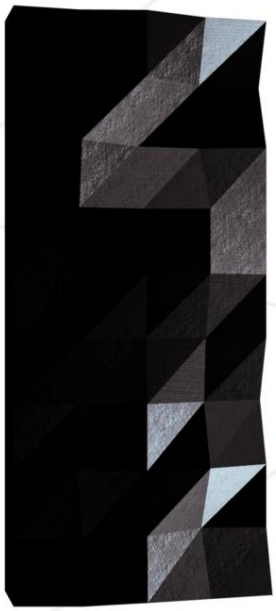

C++ - Módulo 09

STL

Resumo:
Este documento contém os exercícios do Módulo 09 dos módulos C++.

Versão: 2.1

Conteúdo

Capítulo I

Introdução

C++ é uma linguagem de programação de propósito geral criada por Bjarne Stroustrup como uma extensão da linguagem de programação C, ou "C com Classes" (fonte: Wikipedia).

O objetivo destes módulos é apresentar a **Programação Orientada a Objetos.**
Este será o ponto de partida da sua jornada C++. Muitas linguagens são recomendadas para aprender POO. Decidimos escolher C++, pois é derivado do seu velho amigo C.
Como esta é uma linguagem complexa e para manter as coisas simples, seu código estará em conformidade com o padrão C++98.

Sabemos que o C++ moderno é muito diferente em muitos aspectos. Então, se você quer se tornar um desenvolvedor C++ proficiente, cabe a você ir além depois do 42 Common Core!

Capítulo II

Regras gerais

Compilando

- Compile seu código com c++ e os sinalizadores -Wall -Wextra -Werror
- Seu código ainda deve compilar se você adicionar o sinalizador -std=c++98

Convenções de formatação e nomenclatura

- Os diretórios dos exercícios serão nomeados desta forma: ex00, ex01, ... , exn
- Nomeie seus arquivos, classes, funções, funções de membro e atributos conforme necessário em as diretrizes.
- Escreva nomes de classes no formato **UpperCamelCase .** Arquivos contendo código de classe serão sempre será nomeado de acordo com o nome da classe. Por exemplo: ClassName.hpp/ClassName.h, ClassName.cpp ou ClassName.tpp. Então, se você tiver um arquivo de cabeçalho contendo a definição de uma classe "BrickWall" representando uma parede de tijolos, seu nome será BrickWall.hpp.
- A menos que especificado de outra forma, todas as mensagens de saída devem ser encerradas por uma nova linha caractere e exibido na saída padrão.
- *Adeus Norminette!* Nenhum estilo de codificação é imposto nos módulos C++. Você pode seguir seu favorito. Mas tenha em mente que um código que seus avaliadores pares não conseguem entender é um código que eles não conseguem classificar. Faça o seu melhor para escrever um código limpo e legível.

Permitido/Proibido

Você não está mais codificando em C. Hora de C++! Portanto:

- Você tem permissão para usar quase tudo da biblioteca padrão. Assim, em vez de ficar preso ao que você já sabe, seria inteligente usar o máximo possível as versões C++-ish das funções C com as quais você está acostumado.
- No entanto, você não pode usar nenhuma outra biblioteca externa. Isso significa que as bibliotecas C++11 (e formas derivadas) e Boost são proibidas. As seguintes funções também são proibidas: *printf(), *alloc() e free(). Se você usá-las, sua nota será 0 e pronto.

• Observe que, a menos que explicitamente indicado de outra forma, o uso do namespace <ns_name> e palavras-chave de amigos são proibidas. Caso contrário, sua nota será -42.

• **Você tem permissão para usar o STL somente no Módulo 08 e 09.** Isso significa: nenhum **Container** (vetor/lista/mapa/e assim por diante) e nenhum **Algoritmo** (qualquer coisa que exija incluir o cabeçalho <algoritmo>) até então. Caso contrário, sua nota será -42.

Alguns requisitos de design

• Vazamento de memória ocorre em C++ também. Quando você aloca memória (usando o novo palavra-chave), você deve evitar **vazamentos de memória.**

• Do Módulo 02 ao Módulo 09, suas aulas devem ser elaboradas na **Língua Ortodoxa Forma canônica, exceto quando explicitamente indicado de outra forma.**

• Qualquer implementação de função colocada em um arquivo de cabeçalho (exceto para modelos de função) significa 0 para o exercício.

• Você deve ser capaz de usar cada um dos seus cabeçalhos independentemente dos outros. Assim, eles devem incluir todas as dependências de que precisam. No entanto, você deve evitar o problema de inclusão dupla adicionando **guardas de inclusão.** Caso contrário, sua nota será 0.

Leia-me

• Você pode adicionar alguns arquivos adicionais se precisar (por exemplo, para dividir seu código). Como essas atribuições não são verificadas por um programa, sinta-se à vontade para fazê-lo, desde que entregue os arquivos obrigatórios.

• Às vezes, as diretrizes de um exercício parecem curtas, mas os exemplos podem mostrar requisitos que não estão explicitamente escritos nas instruções.

• Leia cada módulo completamente antes de começar! Sério, faça.

• Por Odin, por Thor! Use seu cérebro!!!

Você terá que implementar muitas classes. Isso pode parecer tedioso, a menos que você consiga criar um script no seu editor de texto favorito.

Você tem uma certa liberdade para completar os exercícios. No entanto, siga as regras obrigatórias e não seja preguiçoso. Você iria perca muitas informações úteis! Não hesite em ler sobre conceitos teóricos.

Capítulo III

Regras específicas do módulo

É obrigatório utilizar os recipientes padrão para realizar cada exercício deste módulo.

Depois que um contêiner é usado, você não pode usá-lo para o restante do módulo.

É aconselhável ler o assunto na íntegra antes de fazer o exercícios.

Você deve usar pelo menos um recipiente para cada exercício com o exceção do exercício 02 que exige a utilização de dois recipientes.

Você deve enviar um Makefile para cada programa que compilará seus arquivos de origem para a saída necessária com os sinalizadores -Wall, -Wextra e -Werror.

Você deve usar c++, e seu Makefile não deve ser revinculado.

Seu Makefile deve conter pelo menos as regras $(NAME), all, clean, fclean e re.

Capítulo IV

Exercício 00: Bitcoin Exchange

	Exercício: 00
Troca de Bitcoin	
Diretório de entrega: ex00/	
Arquivos para entrega: Makefile, main.cpp, BitcoinExchange.{cpp, hpp}	
Funções proibidas: Nenhuma	

Você precisa criar um programa que exiba o valor de uma certa quantia de bitcoin em uma determinada data.

Este programa deve utilizar um banco de dados em formato csv que representará o preço do bitcoin ao longo do tempo. Este banco de dados é fornecido com este assunto.

O programa tomará como entrada um segundo banco de dados, armazenando os diferentes preços/datas a serem avaliados.

Seu programa deve respeitar estas regras:

- O nome do programa é btc.
- Seu programa deve receber um arquivo como argumento.
- Cada linha neste arquivo deve usar o seguinte formato: "data | valor".
- Uma data válida sempre estará no seguinte formato: Ano-Mês-Dia.
- Um valor válido deve ser um float ou um inteiro positivo, entre 0 e 1000.

Você deve usar pelo menos um contêiner em seu código para validar este exercício. Você deve lidar com possíveis erros com um apropriado mensagem de erro.

Aqui está um exemplo de um arquivo input.txt:

```
$> head input.txt data | valor
2011-01-03 | 3
2011-01-03 | 2 2011-01-03
| 1 2011-01-03 | 1.2
2011-01-09 | 1 2012-01-11
| -1 2001-42-42 2012-01-11 |
1 2012-01-11 |
2147483648 $>
```

Seu programa usará o valor no seu arquivo de entrada.

Seu programa deve exibir na saída padrão o resultado do valor multiplicado pela taxa de câmbio de acordo com a data indicada em seu banco de dados.

Se a data usada na entrada não existir no seu BD, então você deve usar a data mais próxima contida no seu BD. Tenha cuidado para usar a data mais baixa e não a mais alta.

A seguir está um exemplo de uso do programa.

```
$> ./btc Erro:
não foi possível abrir o arquivo. $> ./btc input.txt
2011-01-03 => 3 = 0.9 2011-01-03
=> 2 = 0.6

03/01/2011 => 1 = 0,3 03/01/2011 =>
1,2 = 0,36
2011-01-09 => 1 = 0,32
Erro: não é um número positivo.
Erro: entrada incorreta => 2001-42-42 2012-01-11 => 1
= 7.1
Erro: número muito grande. $>
```

Aviso: Os contêineres usados para validar este exercício não poderão mais ser usados no restante deste módulo.

Capítulo V

Exercício 01: Notação Polonesa Reversa

	Exercício: 01
RPN	
Diretório de entrega: ex01/	
Arquivos para entrega: Makefile, main.cpp, RPN.{cpp, hpp}	
Funções proibidas: Nenhuma	

Você deve criar um programa com estas restrições:

- O nome do programa é RPN.
- Seu programa deve tomar uma expressão matemática polonesa invertida como argumento.
  mento.
- Os números usados nesta operação e passados como argumentos serão sempre menores que 10. O
  cálculo em si, mas também o resultado, não levam em consideração esta regra.
- Seu programa deve processar esta expressão e gerar o resultado correto no
  saída padrão.
- Caso ocorra um erro durante a execução do programa, uma mensagem de erro deverá ser exibida.
  exibido no erro padrão.
- Seu programa deve ser capaz de manipular operações com estes tokens: "+ - / *".

Você deve usar pelo menos um contêiner em seu código para validar isso
exercício.

Você não precisa gerenciar os colchetes ou números decimais.

Aqui está um exemplo de uso padrão:

```
$> ./RPN "8 9 * 9 - 9 - 4 - 1 +" 42 $> ./RPN "7 7 * 7 -" 42 $> ./

RPN "1 2 * 2/2 * 2 4 - +" 0 $> ./RPN
"(1
+ 1)"


Erro $>
```

Aviso: O(s) contêiner(es) que você usou no exercício anterior são proibidos aqui. O(s) contêiner(es) que você usou para validar este exercício
não poderá ser usado no restante deste módulo.

Capítulo VI

Exercício 02: PmergeMe

	Exercício: 02
PmergeMe	
Diretório de entrega: ex02/	
Arquivos para entrega: Makefile, main.cpp, PmergeMe.{cpp, hpp}	
Funções proibidas: Nenhuma	

Você deve criar um programa com estas restrições:

• O nome do programa é PmergeMe.

• Seu programa deve ser capaz de usar uma sequência de inteiros positivos como argumento.

• Seu programa deve usar o algoritmo de classificação de mesclagem e inserção para classificar os inteiros positivos sequência.

Para esclarecer, sim, você precisa usar o algoritmo Ford-Johnson.

• Se ocorrer um erro durante a execução do programa, uma mensagem de erro deverá ser exibida no erro padrão.

Você deve usar pelo menos dois contêineres diferentes em seu código para validar este exercício. Seu programa deve ser capaz de manipular pelo menos 3000 inteiros diferentes.

É altamente recomendável implementar seu algoritmo para cada contêiner e, assim, evitar usar uma função genérica.

Aqui estão algumas diretrizes adicionais sobre as informações que você deve exibir linha por linha na saída padrão:

- Na primeira linha você deve exibir um texto explícito seguido do positivo não classificado sequência inteira.
- Na segunda linha você deve exibir um texto explícito seguido do positivo classificado sequência inteira.
- Na terceira linha você deve exibir um texto explícito indicando o tempo usado pelo seu algoritmo especificando o primeiro contêiner usado para ordenar o inteiro positivo sequência.
- Na última linha você deve exibir um texto explícito indicando o tempo usado pelo seu algoritmo, especificando o segundo contêiner usado para ordenar a sequência de inteiros positivos.

O formato de exibição do tempo utilizado para realizar a sua triagem é livre, mas a precisão escolhida deve permitir ver claramente o
diferença entre os dois recipientes utilizados.

Aqui está um **exemplo** de uso padrão:

```
$> ./PmergeMe 3 5 9 7 4 Antes: 3 5 9 7 4
Depois: 3 4 5 7 9 Tempo para
processar um intervalo de 5
elementos com std::[..] : 0,00031 us Tempo para processar um intervalo de 5 elementos com std::[..] : 0,00014 us $> ./
PmergeMe `shuf -i 1-100000 -n 3000 | tr "\n" " "` Antes: 141 79 526 321 [...]


Depois: 79 141 321 526 [...]
Tempo para processar um intervalo de 3000 elementos com std::[..] : 62.14389 us Tempo para processar um intervalo de 3000
elementos com std::[..] : 69.27212 us $> ./PmergeMe "-1" "2"

Erro $> #
Para USUÁRIO OSX: $> ./PmergeMe
`jot -r 3000 1 100000 | tr '\n' ' ' [...] $>
```

A indicação da hora é deliberadamente estranha neste exemplo.
É claro que você tem que indicar o tempo usado para executar todas as suas operações, tanto a parte de
classificação quanto a parte de gerenciamento de dados.

Aviso: O(s) recipiente(s) que você usou nos exercícios anteriores são
proibido aqui.

A gestão de erros relacionados com duplicados é deixada ao seu critério.
critério.

Capítulo VII

Submissão e avaliação por pares

Entregue sua tarefa no seu repositório Git como de costume. Apenas o trabalho dentro do seu repositório será avaliado durante a defesa. Não hesite em verificar novamente os nomes das suas pastas e arquivos para garantir que estejam corretos.

16D85ACC441674FBA2DF65190663F33F793984B142405F56715D5225FBAB6E3D6A4F
167020A16827E1B16612137E59ECD492E47AB764CB10B45D979615AC9FC74D521D9
20A778A5E