

Atividades de Paradigmas de Programação 2

Programa de monitoria

Aluno Monitor: Victor Manoel Fernandes de Souza / Professor Orientador: Delair Martinelli

Vale lembrar que aqui vale tudo. Faça uso da STL o máximo que puder. É importante que você resolva a lista de exercícios para melhor absorção do conteúdo.

Antes de começarmos, queria deixar aqui algumas informações

sobre a STL para facilitar seus estudos e seu trabalho. Bem, a Standard Template Library (STL) é uma parte fundamental da linguagem de programação C++. Ela fornece um conjunto de classes e funções que simplificam o desenvolvimento de programas C++, oferecendo estruturas de dados e algoritmos reutilizáveis. Aqui estão algumas das principais bibliotecas STL do C++:

1. Containers (Contêineres):

- vector: Um array dinâmico que redimensiona automaticamente.

- string: armazena sequências de caracteres e fornece uma variedade de operações e métodos para manipulação de strings.

- list: Uma lista duplamente encadeada.

- deque: Uma fila de dupla extremidade (double-ended queue).

- set: Uma coleção de valores únicos, armazenados em ordem.

- map: Uma coleção de pares chave-valor, onde as chaves são únicas e armazenadas em ordem.

- unordered\_set e unordered\_map: Versões não ordenadas de `set` e `map`, que oferecem acesso mais rápido em troca de ordem.

- stack: Uma pilha (LIFO - Last In, First Out).

- queue: Uma fila (FIFO - First In, First Out).

- priority\_queue: Uma fila de prioridade.

2. Algorithms (Algoritmos):

- sort: Ordena uma sequência.

- find: Encontra um valor em uma sequência.

- for\_each: Aplica uma função a cada elemento de uma sequência.

- transform: Aplica uma função e armazena os resultados em outra sequência.

- reverse: Inverte uma sequência.

- max e min: Encontram o valor máximo e mínimo em uma sequência.

- count e count\_if: Contam o número de ocorrências de um valor ou que satisfaçam uma condição.

- accumulate: Soma todos os elementos em uma sequência.

- binary\_search: Realiza busca binária em uma sequência ordenada.

3. Iterators (Iteradores):

- begin e end: Referências ao início e ao final de uma sequência.

- rbegin e rend: Referências ao início e ao final reversos de uma sequência.

- insert: Insere elementos em uma sequência.

- erase: Remove elementos de uma sequência.

4. Functors (Funtores):

- std::function: Permite tratar objetos como funções.

- std::bind: Cria funtores a partir de funções existentes.

5. Utilities (Utilitários):

- pair: Um par de valores.

- tuple: Uma tupla de valores heterogêneos.

- algorithm: Funções utilitárias como `swap`, `min`, `max`, etc.

6. Memory (Memória):

- shared\_ptr e unique\_ptr: Gerenciamento de memória automático e seguro.

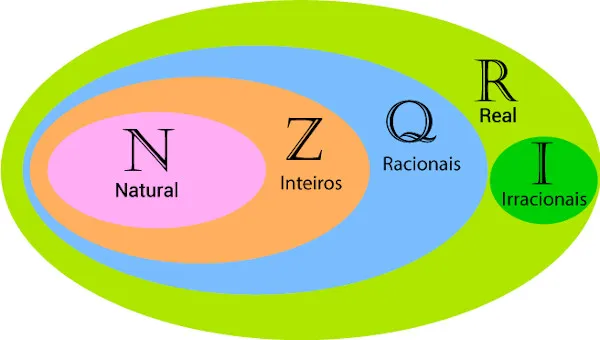
- make\_shared e make\_unique: Criação segura de ponteiros inteligentes.

Estas são algumas das principais bibliotecas da STL em C++ que podem te ajudar no trabalho e exercícios. para mais informações, consulte a documentação no site: <https://cplusplus.com/reference/clibrary/>. É de extrema importância que você se familiarize com a STL, para reaproveitar seu código e evitar trabalho desnecessário.

Como de costume, vamos fazer alguns exercícios para treinar a sintaxe. Faça heranças públicas.

1 - Implemente uma classe genérica para representação de veículos. Todo veículo tem marca e ano. Implemente agora classes derivadas: Carro, Moto e Onibus. Adicione atributos e métodos específicos para essas classes. teste suas funcionalidades.

2 - Conjuntos numéricos são coleções ou agrupamentos de números que compartilham certas propriedades ou características comuns. Eles são usados na matemática para classificar números com base em suas propriedades.



Implemente uma classe para representação do conjunto dos números Reais. Utilizando a classe base implementada, faça classes derivadas representando os Naturais, Inteiros e Racionais. Permita que as classes lance uma exceção caso o número passado não pertença ao conjunto.(use a biblioteca stdexept). teste.

3 - No exercício 7 da lista 1, você implementou a classe Aluno e Professor. refaça o exercício usando o conceito de herança.

4 - Utilizando o exercício 6-a da lista 1, implemente classes derivadas para representar a classe do personagem(Guerreiro, Mago, Elfo, etc). Faça a mesma coisa com os inimigos(escolha seus inimigos).

5 - Utilizando a classe do exercício 4-a como base, implemente uma classe derivada que represente uma pirâmide tridimensional. Sua classe deve calcular o volume. Utilize a mesma ideia no exercício 4-c para um cubo.

6 - Use a classe da questão 16 da lista 1 como classe base, e implemente a classe da questão 17 da lista 1 como classe derivada.

7 - Um grafo simples é um grafo que não contém nem laços nem arestas múltiplas. Um grafo completo é um grafo simples em que todo vértice é adjacente a todos os outros vértices. Utilizando como classe genérica a classe do exercício 15, implemente uma classe que representa grafos Completos, dado o número de vértices.

Vamos nos familiarizar com a STL? vamo q vamo!

8 - Utilizando STL, implemente uma classe que represente:

1. um vetor de inteiros;
2. uma matriz;
3. uma string.

Brinque com as operações disponibilizadas.

9 - Observando a lista 1, percebe-se que, com o uso STL, alguns exercícios poderiam ser resolvidos rapidamente. Identifique TODOS os esses exercícios e refaça-os usando STL.

10 - Em quais situações é interessante o uso de classes derivadas protegida e privada? Pesquise em quais situações aplicar.

11 - Implemente uma classe derivada de std::string, para representar strings. Sua classe deve ter operações de concatenação e verificação se as strings são iguais. As funcionalidade de std::string não podem ser acessíveis via objeto.

12 - Crie uma classe abstrata chamada FormaGeometrica com um método virtual puro chamado calcularArea. Em seguida, crie duas classes derivadas: Circulo e Retangulo, ambas herdando de FormaGeometrica. Implemente o método calcularArea em ambas as classes derivadas para calcular a área de um círculo e de um retângulo. Crie objetos dessas classes e chame o método calcularArea para calcular e exibir as áreas.

13 - Uma forma de se calcular a área de uma figura geométrica complexa, é dividir essa figura em várias figuras simples, como triângulos, retângulos, círculos, etc. Escreva um programa que receba como entrada uma lista de figuras simples (a princípio apenas

triângulos e retângulos) e imprima a área total deste conjunto de figuras. Seu programa deve prever a extensão dos tipos de figuras possíveis (círculos, losangos, etc.). Depois de pronto, faça a extensão para admitir ao menos mais um tipo de figura (sem alterar o código das classes já implementadas). Seu programa deve possuir uma classe responsável por armazenar as figuras e totalizar a soma das áreas.

Obs: não é necessário implementar a leitura das figuras, apenas crie tais figuras e atribua à classe responsável em armazená-las (na função main).

14 - Implemente uma classe base para representar um Animal. A mesma deve ter um método chamado andar. Derivadas dessa classe, construa uma classe para Oviparos e suas características, assim como uma classe Mamiferos e suas características. Ambas as classes devem sobrepor o método andar da classe Animal. Aconteceu de Deus criar um carinha esquisito chamado ornintorrinco. Ele herda características de um ovíparo e de um mamífero. Construa uma classe chamada de Ornitorrinco, onde a mesma deve ser derivada da Classe Oviparo e Mamiferos. Teste a classe Ornitorrinco chamando o método mover.(Leia o exercício 15).

15 - Se você anda atento nas aulas de PP, vai perceber que temos um problema catastrófico com o exercício anterior. Qual é o problema? Como podemos resolvê-lo? Implemente a solução.

16 - (UM CLÁSSICO) Crie um aplicativo, em C++, para a representação de sólidos em R³ e cálculo da área de superfície e volume desses sólidos. Na hierarquia de classes, obrigatoriamente deverá existir uma classe responsável pelo armazenamento do conjunto de objetos, que representam sólidos, e cálculo, de área e volume, total desse conjunto de objetos. Seu aplicativo deve admitir que o usuário realize a inserção de esferas e cilindros, requisitar o cálculo da área e o volume total dos sólidos inseridos. Além disso, deve prever a ampliação do conjunto de sólidos, implementados inicialmente, sem a alteração das classes criadas inicialmente.(estude bem esse exercício, geralmente cai algo parecido).

17 - Usando o código da questão 16, sem alterar nenhuma das classes, adicione a possibilidade de usarmos cubos no sistema. É permitido apenas alterar a função main do programa.

18 - Usando o template vector da STL, implemente uma classe para armazenar um conjunto de elementos inteiros ordenados. Sua classe deve conter apenas uma função de inserção, uma função para a leitura do i-ésimo elemento do conjunto e um método de remoção por chave (remova todos os elementos com um determinado valor). Faça com que todas as funcionalidade do template vector sejam inacessíveis via objeto(da sua classe). Escreva uma aplicação para testar as funcionalidade implementadas.

19 - Uma base numérica pode ser definida como uma sequência de caracteres. Por exemplo, a base decimal pode ser descrita por: 0123456789; a base binária por: 01; a base octal: 01234567; uma base genérica equivalente a base quatro: xyzw (em geral representaríamos a base quatro com 0123). Escreva um programa em C++ que leia um valor n e em seguida uma sequência de n linhas, onde, em cada linha, temos duas sequências de caracteres: a primeira descreve uma base numérica Bi e a segunda um número Xi escrito nessa base numérica. A saída do seu programa consiste em:

Para cada Bi uma linha contendo todos os Xj, onde i≠j , convertidos para a base Bi, na sequência em que aparecem na entrada.

20 - (Atenção: leia a questão 20 e 21, antes de resolver essa questão) A empresa FRITOS possui vários empregados e diferentes tipos de calculo de salário. Um empregado pode, por exemplo, receber como salário pelas horas trabalhadas ou por mês (salário fixo). Além disso, para valorizar a capacitação e fixação dos funcionários, a empresa também proporciona alguns adicionais:

- adicional de capacitação (valor fixo);

- adicional por tempo de serviço (porcentagem sobre o total dos rendimentos do funcionário).

Você foi contratado para escrever um programa em C++ para o calculo do total da folha de pagamento.

Você deve considerar funcionários horistas e mensalistas, além de incluir nos cálculos os adicionais de capacitação e tempo de serviço.

Considere que um funcionário é identificado pelo seu nome, recebe apenas um tipo de salário (por mês ou horas trabalhadas) e pode receber nenhum, algum, alguns ou todos os adicionais.

Não é necessário realizar leitura do teclado (pode criar as entradas direto na função int main()).

Obs: você deve ter uma classe responsável em armazenar os funcionários e realizar a totalização da

folha de pagamento.

21 - A empresa FRITOS precisa de uma manutenção no software que você produziu. Será

necessário adicionar funcionários que recebem uma porcentagem do total de vendas realizadas (comissionado). Além disso, a empresa decidiu criar um novo adicional, o funcionário que tem direito a este adicional recebe um valor fixo e também uma porcentagem sobre o total dos seus rendimentos. O problema é que você **perdeu** (o estagiário kk) o código fonte e só encontra o código da função int main() do seu programa e o restante do código pré-compilado. Realize a manutenção requerida. Obs: isso significa que você só pode alterar a função int main() e adicionar novos arquivos com classes e objetos em seu código.

Referências: Provações do passado realizadas pelo professor