Tipos Abstratos de Dados (TADs) / Modularização

1. Você deve alterar o TAD `ListaVetorInteiros` implementado durante a aula (slides sobre TADs disponíveis no PVANet Moodle) para que ele suporte duas novas operações relacionadas à remoção de elementos da lista de inteiros: `remover_primeiro` e `remover_ultimo`, e uma terceira operação `inverte` que modifica a ordem dos elementos da lista (ou seja [1,2,3] para [3,2,1]).

   Utilize a função `main` abaixo para testar suas funções:

```
int main() {
  ListaVetorInteiros l1;
  l1.inserir_elemento(7);
  l1.inserir_elemento(10);
  l1.inserir_elemento(5);
  l1.inserir_elemento(2);
  l1.inserir_elemento(1);

  l1.imprimir();
  // 7, 10, 5, 2, 1

  l1.remover_primeiro();
  l1.imprimir();
  // 10, 5, 2, 1

  l1.remover_ultimo();
  l1.imprimir();
  //10, 5, 2

  l1.inverte();
  l1.imprimir();
  //2, 5, 10

  return 0;
}
```

2. Implemente um TAD `Esfera` para representação da esfera apresentada na Figura abaixo.

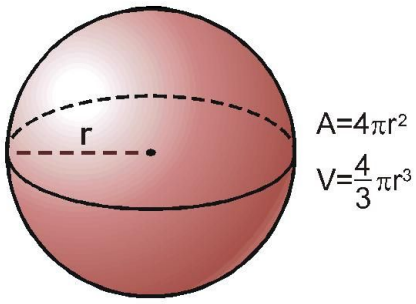

Figura 1: Representação de uma esfera.

Você deve incluir um construtor, um destrutor e uma operação que retorne o seu raio, a sua área e o seu volume. *Importante!* A alocação da esfera deve ser feita dinamicamente. A função que retorna o raio deve ser acionada a partir das funções para cálculo da área e volume da esfera.

Utilize a função `main` abaixo para testar suas funções:

```
int main() {
    float area, volume;
    Esfera *esfera = new Esfera(2.75);
    area = esfera-> calculaAreaEsfera();
    std::cout << area << std::endl;
    volume = esfera-> calculaVolumeEsfera();
    std::cout << volume << std::endl;
    delete esfera;

    return 0;
}
```

3. Implemente um TAD `ContaBancaria` que contenha minimamente os seguintes atributos: `nome_titular`, `numero_conta`, `saldo_inicial`. Você deve implementar um construtor, métodos de acesso (`get`), um método `depositar`, `sacar`, que realiza um saque a partir de um valor disponível na conta, e `exibirInformacoes`, que imprime informações de uma conta específica.

```
int main() {
    double valor;
    ContaBancaria conta("Maria Silva", 12345, 1000.0);
    conta.exibirInformacoes();
    std::cin >> valor;
    conta.depositar(valor);
    conta.getsaldo();
    if (conta.sacar(valor)) {
        std::cout << "Saque realizado.\n";
    } else {
        std::cout << "Saldo insuficiente.\n";
    conta.exibirInformacoes();

    return 0;
}
```

4. Implemente um TAD `Aluno` que contenha minimamente os seguintes atributos: `nome`, `matrícula`, `notas`, `media_notas`. É importante ressaltar que o atributo `notas` está relacionado à quantidade de matérias cursadas pelo aluno. Ou seja, se o aluno cursa 10 matérias, o programa deve armazenar às 10 notas equivalentes de forma dinâmica (1 nota para cada matéria). Você deve implementar um construtor para inicialização das variáveis. No caso do atributo `media_notas`, ele deve ser inicializado com 0 (zero) até os respectivos valores de notas sejam lidos. Você deve implementar um método `adicionarNota` responsável pelo preenchimento do vetor de notas do aluno – neste caso você pode utilizar uma variável auxiliar para controle dos índices do vetor de notas. Além disso, você deve implementar um método `calcularMediaAluno`, que recebe como argumento as notas e retorna a média dos valores. Por fim, você deve implementar um método `imprimeInformacoesAluno`, que imprime as informações do aluno.

Considerações Gerais!

- Exercício individual.
- Entrega: conforme agendado no PVANET Moodle;
- Conforme estrutura abaixo apresentada crie um projeto para resolução de cada exercício (ex.: pratica4_exercicio1.zip, pratica4_exercicio2.zip). Cada projeto deve conter os arquivos `.h`, `.cpp`, e `main.cpp` criados para resolução do exercício. Envie, através do PVANet Moodle, uma pasta compactada (.rar ou .zip) contendo todos os projetos (também compactados). A pasta compactada deve conter informações do aluno (ex.: julio_reis-pratica4.zip).

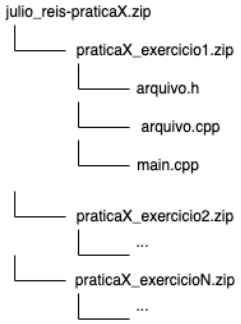

Figura 2: Estrutura de diretórios.

- O seu `main.cpp` deve conter, minimamente, instruções para criação (instanciação de objetos) e chamadas das funções implementadas (TODAS!!!). Para teste, você pode usar os exemplos fornecidos.