Tipos Abstratos de Dados (TADs)

1. Implemente um TAD `Contador` que gerencie um número inteiro, com a capacidade de incrementar, decrementar e exibir o valor atual do contador. Seu deve deve possuir um construtor que inicializa `valor` contador com o valor fornecido. Além disso, o seu deve conter os métodos `incrementar`, que incrementa o valor do contador em 1; `decrementar`, que decrementa o valor do contador em 1 (o valor não deve ser decrementado abaixo de zero), `imprimirValor`, que imprime o valor atual do contador, e por fim, `resetarValor`, que define o valor do contador para zero.

```
int main() {
    Contador c(1);
    c.incrementar();
    c.imprimirValor();
    c.resetarValor();
    c.decrementar();

    return 0;
}
```

2. Implemente um TAD `Horario` com três atributos inteiros: `hora`, `minuto` e `segundo`. Faça um construtor que inicializa as três variáveis. O seu TAD deve ser robusto para suportar a representação do tempo tanto no formato 24h ex: (13:35:15) quanto no formato 12h (01:35:15 PM). Para isso, você deve implementar o método `exibeHorarioUniversal`, que imprime o tempo em formato universal (24h) e o método `exibeHorarioPadrao` que imprime o tempo em formato padrão (12h).

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

```
int main() {
    Horario h(13, 35, 15);
    h.exibeHorarioUniversal();
    h.exibeHorarioPadrao();

    return 0;
}
```

3. Implemente um TAD `Data` com três atributos inteiros: `dia`, `mês` e `ano`. Faça um construtor que inicializa as três variáveis e suponha que os valores passados serão sempre corretos. O TAD deve possuir um método para exibir a data em formato de números separados por barra: dia/mes/ano e outro método para exibir a data por extenso (ex: 28 de fevereiro de 2021). Por fim, você deve implementar um método `obterDataEmSegundos` que a partir de uma data recebida como argumento retorna sua representação em segundos (em caso de dúvidas, buscar por *unix timestamp* - vale lembrar que a data inicial considerada será 01/01/1970 e que para fins de simplificação não vamos considerar o horário no cálculo, o padrão será sempre 00:00:00).

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

```
int main() {
    Data d1(13, 12, 2021);
    d1.exibirDataFormatoNumero();
    d1.exibirDataPorExtenso();
    d1.obterDataEmSegundos();

    return 0;
}
```

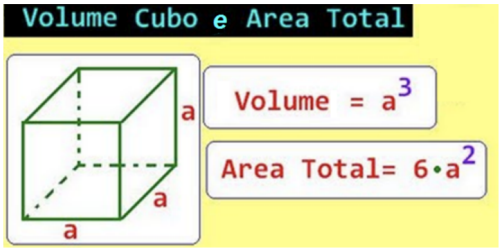

Figura 1: Representação de um cubo.

4. Implemente um TAD `Cubo` para representação do cubo apresentado na Figura 1.

   Você deve implementar um construtor e as operações cálculo da área e do volume do cubo. *Importante!* A alocação do cubo deve ser feita dinamicamente.

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

```
int main() {
    float area, volume;
    Cubo *cubo = new Cubo(3.0);
    area = cubo-> calculaAreaCubo();
    std::cout << area << std::endl;
    volume = cubo-> calculaVolumeCubo();
    std::cout << volume << std::endl;
    delete cubo;

    return 0;
}
```

Considerações Gerais!

- Exercício individual.
- Entrega: conforme agendado no PVANET Moodle;
- Conforme especificado crie um projeto para resolução de cada exercício (ex.: pratica3_exercicio1.cpp, pratica3_exercicio2.cpp, etc). Envie, através do PVANet Moodle, uma pasta compactada (.rar ou .zip) contendo todos os projetos. A pasta compactada deve conter informações do aluno (ex.: julio_reis-pratica3.zip).