Algoritmos e Programação II

VI Lista de Exercícios: Introdução a Classes

Prof. Evandro C. R. Rosa
UNIVALI

Nome Completo: _____ Código de Aluno: _____

1. Responda sucintamente:

   (a) Qual é a diferença entre uma classe e uma instância da classe?

   (b) Qual é a diferença entre a estrutura `Pessoa` e a classe `Pessoa`, como mostrado abaixo?

```
struct Pessoa {
    string nome;
    int idade;
};
```

```
class Pessoa {
    string nome;
    int idade;
};
```

   (c) Qual é a especificação de acesso padrão dos membros de uma classe?

   (d) Veja a definição da seguinte função membro:

```
void Circulo::obterRaio()
```

      i. Qual é o nome da função?

      ii. De qual classe a função é membro?

   (e) É uma boa ideia tornar as variáveis membros privadas?

   (f) Em quais circunstâncias uma função membro deve ser privada?

   (g) O que é um construtor? O que é um destrutor?

   (h) O que é um construtor padrão? É possível ter mais de um construtor padrão?

   (i) É possível ter mais de um construtor? E mais de um destrutor?

2. Os códigos abaixo contêm um ou mais erros. Indique quais são:

   (a)

```
class Circulo: {
    private
        double centroX;
        double centroY;
        double raio;
    public
        definirCentro(double, double);
        definirRaio(double);
}
```

(b)
```
#include <iostream>
using namespace std;

class Haltere;
{
    int peso;

  public:
    void definirPeso(int);
};

void definirPeso(int p) { peso = p; }

int main() {
    Haltere barra;
    Haltere(200);
    cout << "O peso é " << barra.peso << endl;
    return 0;
}
```

(c)
```
class Troco {
  public:
    int centavos;
    int cinco_centavos;
    int dez_centavos;
    int vinte_cinco_centavos;
    Troco() {
        centavos = cincoCentavos = dezCentavos = vinteCincoCentavos = 0;
    }
    Troco(int c = 100, int cc = 50, d = 50, vcc = 25);
};

void Troco::Troco(int c, int cc, d, vcc) {
    centavos = c;
    cincoCentavos = cc;
    dezCentavos = d;
    vinteCincoCentavos = vcc;
}
```

3. Crie uma classe chamada `Data`. A classe deve armazenar uma data em três inteiros: mês, dia e ano. Devem haver funções membros para imprimir a data nos seguintes formatos:

   - 25/12/2024
   - 25 de dezembro de 2024

   Demonstre a classe escrevendo um programa completo que a implemente. Validação de entrada: Não aceite valores para o dia maiores que 31 ou menores que 1. Não aceite valores para o mês maiores que 12 ou menores que 1.

4. Crie uma classe chamada `Estoque` que armazene informações e calcule dados sobre itens no inventário de uma loja. A classe deve ter as seguintes variáveis privadas:

Nome da Variável	Descrição
`codigo_item`	Um int que armazena o código do item.
`quantidade`	Um int que armazena a quantidade de itens em estoque.
`custo`	Um double que armazena o custo por unidade do item.
`custo_total`	Um double que armazena o custo total do item (calculado como quantidade vezes custo).

A classe deve ter as seguintes funções públicas:

- Construtor Padrão: Define todas as variáveis membro como 0.
- Construtor: Aceita como argumentos o código, o custo e a quantidade de um item. A função deve copiar esses valores para as variáveis membro apropriadas e então chamar a função `definir_custo_total`.
- `definir_codigo_item`: Aceita um argumento do tipo inteiro que é copiado para a variável membro `codigo_item`.
- `definir_quantidade`: Aceita um argumento do tipo inteiro que é copiado para a variável membro quantidade.
- `definir_custo`: Aceita um argumento do tipo double que é copiado para a variável membro custo.
- `definir_custo_total`: Calcula o custo total do item no inventário (quantidade vezes custo) e armazena o resultado em `custo_total`.
- `obter_codigo_item`: Retorna o valor de `codigo_item`.
- `obter_quantidade`: Retorna o valor de quantidade.
- `obter_custo`: Retorna o valor de custo.
- `obter_custo_total`: Retorna o valor de `custo_total`.

Demonstre a classe em um programa.

Validação de Entrada: Não aceite valores negativos para código, quantidade ou custo.

5. Crie uma classe chamada `NotasProva` que tenha variáveis membro para armazenar três notas de prova. A classe deve ter um construtor, funções de acesso e de modificação para as notas, e uma função membro que retorne a média das notas. Demonstre a classe escrevendo um programa separado que crie uma instância da classe e peça ao usuário para inserir três notas de prova, que devem ser armazenadas no objeto `NotasProva`. O programa deve exibir a média das notas, conforme relatado pelo objeto `NotasProva`.

6. Escreva uma classe chamada Círculo que tenha as seguintes variáveis de membro:

   - `raio`: um `double`
   - `pi`: um `double` inicializado com o valor 3.14159

   A classe deve ter as seguintes funções membro:

   - Construtor Padrão: um construtor padrão que define o raio como 0.0.
   - Construtor: aceita o raio do círculo como argumento.
   - `set_raio`: uma função mutadora para a variável raio.
   - `get_raio`: uma função acessora para a variável raio.
   - `get_area`: retorna a área do círculo, calculada como área = pi $\times$ raio^2.

- `get_diametro`: retorna o diâmetro do círculo, calculado como diâmetro = 2raio.
- `get_circunferencia`: retorna a circunferência do círculo, calculada como circunferência = 2pi × raio.

Escreva um programa que demonstre a classe Círculo, solicitando ao usuário o raio do círculo, criando um objeto Círculo e, em seguida, exibindo a área, o diâmetro e a circunferência do círculo.

7. Projete uma classe que tenha um array de números de ponto flutuante. O construtor deve aceitar um argumento inteiro e alocar dinamicamente o array para armazenar essa quantidade de números. O destrutor deve liberar a memória ocupada pelo array. Além disso, deve haver funções membro para realizar as seguintes operações:

   - Armazenar um número em qualquer elemento do array
   - Recuperar um número de qualquer elemento do array
   - Retornar o maior valor armazenado no array
   - Retornar o menor valor armazenado no array
   - Retornar a média de todos os números armazenados no array

   Demonstre a classe em um programa.

8. Escreva uma classe chamada `Moeda` que deve ter a seguinte variável membro:

   - Uma string chamada `lado_para_cima` que deve armazenar "cara" ou "coroa", indicando qual lado da moeda está voltado para cima.

   A classe Moeda deve ter as seguintes funções membro:

   - Um construtor padrão que determina aleatoriamente qual lado da moeda está voltado para cima ("cara" ou "coroa") e inicializa a variável `lado_para_cima` de acordo.
   - Uma função membro `jogar` que simula o lançamento da moeda. Quando a função `jogar` é chamada, ela determina aleatoriamente qual lado da moeda está voltado para cima e define a variável `lado_para_cima` de acordo.
   - Uma função membro chamada `get_lado_para_cima` que retorna o valor da variável `lado_para_cima`.

   Escreva um programa que demonstre a classe Moeda. O programa deve criar uma instância da classe e exibir o lado que está inicialmente voltado para cima. Em seguida, use um loop para lançar a moeda 20 vezes. Cada vez que a moeda for lançada, exiba o lado que está voltado para cima. O programa deve contar o número de vezes que "cara" está voltado para cima e o número de vezes que "coroa" está voltado para cima, e exibir esses valores após o loop.