Universidade Federal de São Carlos

ORIENTAÇÃO A OBJETOS: CLASSES

Disciplina: Programação de Computadores

Data de entrega: até 20 de setembro de 2018.

Professor: Delano Medeiros Beder

1 Enunciado

A atividade T1 consiste em implementar em C++ as classes conforme descritas abaixo:

1. [2,50] Defina a classe Aluno cujos objetos representam alunos matriculados em uma disciplina. Cada objeto dessa classe deve guardar os seguintes dados do aluno: RA, nome, nota da 1^a prova, nota da 2^a prova e nota do trabalho. Os atributos da classe devem ser **privados**. Escreva os seguintes métodos/construtores para esta classe:

Nome	Descrição
Aluno(int, string, double, double, double)	Construtor capaz de setar os atributos do objeto. Esse construtor deve ser único.
int getRA()	Método responsável por retornar o RA do aluno.
string getNome()	Método responsável por retornar o nome do aluno.
double media()	Método responsável por calcular a média final (MF) do aluno – cada prova tem peso 7 e o trabalho tem peso 6.
bool aprovado()	Método responsável por retornar verdadeiro se o aluno foi aprovado ($MF \geq 6.0$) e falso, caso contrário.
bool sac()	Método responsável por retornar verdadeiro se o aluno ficou em SAC – Sistema de Avaliação Complementar ($5.0 \leq MF < 6.0$) e falso, caso contrário.
double notaSAC()	Método responsável por calcular qual a nota mínima necessária, na prova de avaliação complementar (SAC), para aprovação na disciplina. Para o aluno ser aprovado após a prova de avaliação complementar (SAC) precisa atender a seguinte regra: $\frac{SAC+MF}{2} \geq 6.0$. Caso o aluno não ficou em SAC, retornar 0.

2. [2,00] Defina a classe Data com os dados: dia, mês e ano. Os atributos da classe devem ser **privados**. A classe deverá dispor dos seguintes métodos/construtores:

Nome	Descrição
Data(int, int, int)	Construtor capaz de setar os atributos (dia, mês e ano). Este construtor verifica se a data está correta, caso não esteja, a data é configurada como 01/01/0001. Esse construtor deve ser único.
int compara(Data&)	Método que recebe como parâmetro um outro objeto da Classe Data, compara com a data corrente e retorna: **0** se as datas forem iguais; **1** se a data corrente for maior que a do parâmetro; **-1** se a data do parâmetro for maior que a corrente.
int getDia()	Método responsável por retornar o dia da data.
int getMes()	Método responsável por retornar o mês da data.
string getMesExtenso()	Método responsável por retornar o mês da data corrente por extenso (Janeiro, Fevereiro, Março, Abril, Maio, Junho, Julho, Agosto, Setembro, Outubro, Novembro e Dezembro).
int getAno()	Método responsável por retornar o ano da data.
void imprime()	Método responsável pela impressão das informações de uma data (no formato DD/MM/YYYY).

3. [3,00] Defina a classe Voo cujos objeto representam vôos que acontecem em determinada data e em determinado horário. Cada objeto dessa classe deve guardar os seguintes dados do vôo: número, data e horário. Cada vôo possui no máximo 100 passageiros, e a classe permite controlar a ocupação das vagas. Os atributos da classe devem ser **privados**. A classe deve ter os seguintes construtores/métodos:

Nome	Descrição
Voo(int, Data, string)	Construtor capaz de setar os atributos: número do vôo, data (instância da classe Data definida na questão anterior) e horário.
int proximoLivre()	Método responsável por retornar o número da próxima poltrona livre. Retorna zero se não houver poltrona disponível no vôo.
bool verifica(int)	Método responsável por verificar se o número da poltrona recebido como parâmetro está ocupada.
bool ocupa(int)	Método responsável por ocupar determinada poltrona do vôo, cujo número é recebido como parâmetro, e retornar verdadeiro se a poltrona não estiver ocupada (operação foi bem sucedida) e falso caso contrário.
bool desocupa(int)	Método responsável por desocupar determinada poltrona do vôo, cujo número é recebido como parâmetro, e retornar verdadeiro se a poltrona estiver ocupada (operação foi bem sucedida) e falso caso contrário.
int vagas()	Método responsável por retornar o número de poltronas vagas disponíveis (não ocupadas) no vôo.
void imprime()	Método responsável pela impressão das informações de um vôo (número, data, horário, quantidade de vagas).
Métodos getters e setters	Métodos responsáveis por ler e alterar cada um dos atributos em separado.

4. [2,50] Você foi contratado por uma agência de viagens para implementar um aplicativo de conversão de reais para dólar de acordo com a taxa de compra e a taxa de venda. Para isso, você deve implementar a classe Conversor que inclui os seguintes atributos (taxa de compra e taxa de venda). Os atributos da classe devem ser **privados**. A classe deve ter os seguintes construtores/métodos:

Nome	Descrição
Conversor(double, double)	Construtor capaz de setar os atributos: taxa de compra e taxa de venda.
void imprimeTaxas()	Método responsável por imprimir o valor das 2 taxas de conversão.
double vendeDolar(double)	Método que recebe uma quantia em dólares e devolve o valor correspondente em reais.
double compraDolar(double)	Método que recebe uma quantia em reais e devolve o valor correspondente em dólares.
Métodos getters e setters	Métodos responsáveis por ler e alterar cada um dos atributos em separado.

2 Observações importantes

2.1 Sobre a elaboração e entrega:

- Este exercício-programa deve ser elaborado individualmente.
- Você deve utilizar **apenas** os conceitos apresentados em aula.
- Você deve implementar as classes em C++.
- Compacte o código-fonte das classes em C++ (e se possível, o projeto Netbeans ou CodeBlocks) e entregue somente este arquivo no ambiente moodle. O arquivo <RA>.zip deve conter o código-fonte das classes em C++.

  Exemplo: 1234567.zip (cuidado para não enviar arquivos errados!)

- O prazo de entrega é o dia 20 de setembro de 2018 às 23h55.
- A entrega será feita unicamente pelo ambiente moodle (`https://ava.ead.ufscar.br`). Não serão aceitos trabalhos enviados por email.
- Guarde uma cópia dos arquivos entregues.

2.2 Sobre a avaliação:

- Não serão toleradas cópias! Exercícios copiados (com ou sem eventuais disfarces) receberão nota ZERO. O exercício do aluno alvo da cópia também receberá nota ZERO.
- Exercícios com erros de sintaxe (ou seja, erros de compilação) receberão nota ZERO.
- Os exercícios serão avaliados segundo os seguintes critérios:
  - Soma simples dos valores obtidos nos itens de 1 a 2
    1. Atendimento às normas de boas práticas de programação (comentários, endentação, nomes de variáveis, estruturação do código, etc) [0..20]
    2. Corretude na implementação da atividade [0..80]