Data: 10/05/2018 – 1º Semestre de 2017.
Prof. Dr. Jean M. Simão - DAINF/UTFPR Curitiba-Central.

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ - Campus de Curitiba Central – **Dep. Acadêmico de Informática** (DAINF). Disciplina: **Técnicas de Programação – CSE20**. **Prova sobre linguagem C++ / Diagrama de Classes (*UML*) / Orientação a Objetos - 1ª Parcial**

Nome do Aluno: _____ **Turma**: S02
Curso: _____ Horário de Começo: _____ **Horário de Fim**: _____

Leia toda a prova, pois os enunciados estão completados uns nos outros.
Utilize os bons princípios de projeto e programação orientada a objetos.
Em tempo, questões de mesmo peso, idem para os itens dentro delas (quando houver).

(Questão - 1) (a) Em um programa C++ (*console*) para simulação de trânsito, crie uma classe *Veiculo* que terá um atributo protegido *string* chamado *nome*, bem como outros atributos protegidos *float* chamados de *x* e *y* com seus respectivos *get*s. **(b)** Esta classe deve ter um método virtual chamado *atualizar* cuja implementação *default* atualiza *x* e y em 0.5 cada qual. Ainda, esta classe deve ter um método *mostrar* que chama *atualizar* e daí mostra os valores de *nome*, *x* e *y*. **(c)** Essa classe permitirá duas classes derivadas dela: *Carro* e *Viatura*. Essas duas classes terão cada qual um atributo protegido, respectivamente inteiro *multas* e booleano *sirene*. **(d)** Nessas classes requisitadas, assim como nas das demais questões, deve-se usar *const* sempre que possível.

(Questão - 2) (a) O método *atualizar* será redefinido em *Carro* de maneira tal a permitir que se possa incrementar o *x* em 1.0 e o *y* em 1.0, bem como será redefinido em *Viatura* para permitir que se possa incrementar o *x* em *algo* (i.e., algum valor real) e o *y* em 1.0. Este presente item deve ser feito a luz do próximo item, usando sobrecarga de operadores ++ e = respectivamente nas classes *Carro* e *Viatura*. Até é aceitável que este presente item seja feito de outra forma, sem tal sobrecarga de operadores, mas obviamente sem então cumprir o próximo item que segue. **(d)** No *atualizar* do *Carro* deve-se chamar o operador ++ de forma tal a permitir o incremento em *x* e *y* já determinados nesse contexto. No atualizar de *Viatura* chamar o operador = parametrizado com *algo* de forma tal a permitir o incremento em *x* e *y* já determinados nesse contexto.

(Questão - 3) (a) No contexto do programa C++ em questão, crie uma classe chamada *Frota*, usando o componente *set* (conjunto) da STL, para endereços de objetos que possam ser apontados como *Veiculo*. **(b)** Essa classe deverá ter método de inclusão, lembrando que *set* não permite repetir elementos por definição. **(b)** Essa classe deverá ter método de exclusão de todos os elementos. **(d)** Por fim, essa classe dever ter um método de percorrimento que começa pelo primeiro elemento e vai até último, o qual chama o método *mostrar* de cada objeto apontado enquanto *Veiculo*. **Obs.**: *Alternativamente ao componente set, desde que com as mesmas funcionalidades, pode-se compor a classe Frota usando: (1) lista encadeada. (2) alocação e deslocação dinâmica de vetor, mas valendo 3/4 da questão; (3) outro componente da STL, mas valendo 2/4 da questão; ou (4) vetor usual de tamanho fixo, mas valendo 1/4 da questão*.

(Questão - 4) (a) Crie uma classe *Simulador,* com um *método* executar, sendo que a instância ou o objeto dela (na função *main*) se chamará *CTA*. Em *Simulador* haverá como atributos dois objetos de *Carro* tendo os seguintes valores inicializados via construtora: para o 1º como segue *nome* = "cidadão", *x*=1, *y* =1 e *multas* = 0; para o segundo como segue *nome*="meliante", *x*=4, *y*=1 e *multas*= 500. Em *Simulador* haverá também como atributo uma instância de *Viatura*, tendo os seguintes valores inicializado via construtora *x*=0, *y*=0, *sirene*=true e *algo* 2. **(b)** Em simulador ainda haverá como atributo privado um objeto de *Frota* no qual será inserido todos os endereços de objetos pertinentes usando *cast* apropriado. **(c)** Todos os objetos agregados em *Simulador* devem ser alocados e desalocados dinamicamente. **(d)** No método *executar*, faça um laço de repetição que: (1) mostra a evolução de posição de todas as viaturas via objeto pertinente de *Frota*. (2) para quando a 'polícia' alcançar o 'meliante' ou quando passar de dez iterações o laço.

(Questão - 5) (a) Primeiramente, antes das demais questões, elabore um diagrama em *UML* das classes solicitadas nelas e de seus relacionamentos. **(b)** Ainda, tanto em *UML* quanto em C++, haverá construtora(s) e destrutora apropriados, sento todos os atributos inicializados por elas. **(c)** Por fim, não haverá nenhum *set* ou similar ou métodos complementares outros nas classes. **(d)** Na função *main()*, apenas poderá ser instanciado *CTA*, o que faz chamar sua construtora.

A interpretação faz parte do conteúdo da prova! Inclua comentários (se for o caso) para deixar explicitas suas decisões.