Lista 2: Nivelamento – Arquivos e Introdução a Programação por Objetos

Obs: Cópias serão desconsideradas, ou seja, a nota será igual a 0 (zero).

1) Crie as classes conforme a descrição abaixo:

a) Crie a classe Aluno com os atributos privados: nome, do tipo String, idade, do tipo int, peso, do tipo double, formando, do tipo boolean e sexo, do tipo char.

b) Crie o construtor da classe Aluno que recebe parâmetros para inicializar os atributos nome, idade, peso e sexo. O atributo formando deve ser inicializado automaticamente com falso.

c) Crie os métodos de acesso (Get) para todos os atributos.

d) Crie os métodos modificadores (Set) para todos os atributos.

e) Crie a classe TesteAluno com um método Main.

f) No método Main:

- Peça para o usuário entrar com os valores necessários para criar um aluno, instancie um objeto Aluno, depois imprima todos os atributos do objeto Aluno usando os métodos de acesso (Get) de Aluno.
- Depois, pergunte novamente ao usuário os valores da idade, do peso e se o aluno é formando. Altere estes atributos com os parâmetros informados por meio dos métodos modificadores (Set).
- Por fim, imprima novamente os valores dos atributos do objeto Aluno usando os métodos de acesso (Get) de Aluno

2) Crie uma classe Cliente, que represente os clientes de um banco. A classe deve ter os seguintes atributos:

- codigo (tipo int)
- nome (tipo string)
- eClienteEspecial (tipo bool) //Informa se o cliente pertence a categoria Cliente Especial
- limiteCredito (tipo double) //Limite de crédito do cliente
-

a) Escreva um método construtor para a classe. O construtor deve receber como parâmetro o código e o nome do cliente. No construtor o atributo limiteCredito deve receber zero, e o atributo eClienteEspecial deve receber false.
b) Escreva um método GetNome. O método não deve ter parâmetros e deve retornar o atributo nome.
c) Escreva um método GetCodigo. O método não deve ter parâmetros e deve retornar o atributo codigo.
d) Escreva um método AtualizarLimite que atualize o limite de crédito do cliente. O método deverá receber como parâmetro um valor do tipo double. Caso o cliente seja especial, o seu limite de crédito deverá ser atualizado com o valor recebido como parâmetro e o método deverá retornar true, indicando que a operação foi realizada. Caso contrário, o método deverá retornar false.
e) Escreva um método AtualizarCategoria que atualize a categoria do cliente (o seu atributo eClienteEspecial). O método deverá receber como parâmetro um boolean. Além de atualizar o referido atributo com o valor recebido como parâmetro, se este valor for false, o atributo limiteCredito deverá ser atualizado com o valor 0 (zero). Visto que, somente Cliente Especiais podem ter crédito.
f) Crie uma classe Teste que contenha o método Main. No Main, crie três clientes e utilize todos os métodos da classe Cliente.

3) Em uma agenda telefônica os contatos são cadastrados com os seguintes dados:

- nome: nome do contato;
- celular: cadeia de caracteres com o número do celular do contato;
- email: cadeia de caracteres com o e-mail do contato;
- aniversario: data de aniversário do contato (contendo dia e mês).

Escreva um programa em C#, para simular uma agenda telefônica. O programa deve apresentar um menu com as seguintes operações:

1. Inserir um novo contato
2. Buscar um contato pelo nome
3. Imprimir todos os dados de todos os contatos da agenda telefônica
4. Encerrar programa

Para tanto, crie as seguintes classes

Classe **Data**, com atributos dia e mês. Crie um construtor, métodos Get e Set.

Classe **Contato**, com atributos nome, celular, email, aniversario (tipo Data). Crie um método construtor que receberá como parâmetro: nome, celular, email dia do aniversário e mês do aniversário. Crie um outro método construtor que receberá como parâmetro: nome, celular, data do aniversário (tipo Data). Crie os métodos Get e Set.

Classe **AgendaTelefonica**, com atributos: agenda (tipo vetor de Contatos), quant (tipo int). O atributo quant deve armazenar a quantidade de contatos que estão na agenda. Assim, esse atributo deverá ser incrementado a cada nova inserção. A classe **AgendaTelefonica** deverá ter os seguintes métodos:

- Método construtor: deverá receber o tamanho da agenda (int). Deve ser criado o vetor agenda e o atributo quant deve ser inicializado com zero.
- InserirContato: o método deverá receber um Contato como parâmetro e deverá ser inseri-lo no vetor agenda, caso ainda tenha espaço disponível na agenda.
- BuscarContato: o método deverá receber um nome (string) como parâmetro e deverá retornar um objeto Contato, caso o nome conste na agenda. Caso o nome não conste na agenda deverá ser retornado o valor null.
- ImprimirAgenda: método sem parâmetros e sem retorno. O método deverá imprimir os dados de todos os contatos da agenda.

Classe **Teste**, deverá ter o método Main. No Main deve ser instanciado um objeto Agencia Telefonica com tamanho 50. O método deverá apresentar o menu de opções para o usuário. A opção escolhida pelo usuário deverá ser lida e executada. O programa deverá repetir esses passos até que a opção 4 seja escolhida pelo usuário.

4) Implemente a classe Estacionamento com os seguintes atributos:

```
private string nome;
private int numTotalVagas;
private int numVagasLivres;
private string[] vagas;
```

a) Implemente os seguintes métodos:

- Estacionamento(string nome, int numTotalVagas)
  Método construtor que inicializa os atributos nome e numTotalVagas com os valores passados por parâmetro. O atributo numVagasLivres deverá ser inicializado com o número total de vagas do estacionamento. Instancie o vetor vagas;
- int Estacionar(string placa)
  Se existirem vagas disponíveis no estacionamento, encontrar a primeira posição vazia do vetor vagas e inserir a placa informada como parâmetro. O método deve retornar o número da vaga (isto é a posição do vetor que a placa foi inserida). Se não existirem vagas disponíveis, esse método deve retornar -1;
- int ObterVagaOcupada(string placa)
  Retorna o número da vaga, ou seja, a posição do vetor vagas, ocupada pelo veículo com a placa informada como parâmetro. Se a placa informada não for encontrada, deve retornar -1;
- void RetirarVeiculo(string placa)
  Retira o veículo cuja placa corresponde à informada como parâmetro para esse método, marcando a vaga correspondente com null;
- int GetNumVagasLivres()
  Retorna o número de vagas disponíveis no estacionamento;
- void ExibirOcupacaoEstacionamento()
  Imprime o número de cada vaga do estacionamento e a placa do veículo que a ocupa ou a informação de que a vaga está vazia.

b) Implemente a classe Teste, com o método Main e as seguintes manipulações:

- Crie um objeto Estacionamento com 10 vagas
- Estacionar 4 carros com as seguintes placas: HKT0098, OLP4290, HJB0495 e OWB3904
- Exibir a ocupação do estacionamento
- Exibir o número da vaga onde o carro de placa HKT0098 está estacionado.
- Retirar do estacionamento o veículo de placa HKT0098"
- Exibir a ocupação do estacionamento
- Estacionar 4 carros com as seguintes placas: HTP5619, BOL4861 e HGT9436
- Exibir a ocupação do estacionamento
- Exibir a quantidade de vagas livres no estacionamento

5) Faça um programa que leia um arquivo texto contendo uma lista de compras (Crie um arquivo para testar seu programa). Cada linha do arquivo deve ter o seguinte formato:

nomeProduto;quantidade;valorProduto

Exemplo:

Shampoo;2;8,99

O programa deverá exibir na tela o valor total da compra.

6) Faça um programa que apure o resultado de uma votação para determinar o *personagem animado* favorito das pessoas. Suponha que existam cinco candidatos cujos códigos de identificação são:

0. Perna Longa
1. Pluto
2. Mickey
3. Bob Esponja
4. Cebolinha

Considere um arquivo texto (denominado “votos.txt”) que contém, em cada linha, um determinado voto (um voto é representado pelo código de identificação do candidato). O programa deverá apresentar, como resultado, o nome do candidato e a quantidade de votos do candidato mais votado, o código de identificação e a quantidade de votos do candidato menos votado e a quantidade de votos nulos (um voto nulo é um voto cujo código de identificação é um inteiro diferente de 0,1,2,3,4).

Dica: crie vetores para fazer o processamento