TRABALHO PRÁTICO 3

MODALIDADE:	Aprendizagem +	Escolha um item.	
CURSO:	Tecnologias de Programação de Sistemas de Informação		
UC:	Desenvolver programas complexos em linguagem de programação estruturada	CÓDIGO UC:	UC00607
FORMADOR/A:	Bruno Silva	DATA:	
FORMANDO/A:	WESLEY FILIPE FERREIRA DE SOUZA	CLASSIFICAÇÃO:	

Para este trabalho prático deve criar um novo projeto com o nome "**avaliacao_3_primeiro_ultimo",** onde as palavras **primeiro** e **ultimo** devem ser **substituídas** pelo vosso primeiro e último nome.

O objetivo é desenvolver um programa para gerir um programa adaptado a classe de cada formando/a. Deve efetuar os seguintes passos:

1. Criar a pasta com o nome "**classes**" e adicionar a classe com o nome **Hospital.cs** com os seguintes conteúdos:

   Variáveis de instância

   ID

   Nome_Hospital

   Localizacao

   Numero_Estacionamentos

   Numero_Camas

   Numero_Profissionais

   Construtor da Classe (com os **dados referenciados mais acima)**

   Métodos de Instância (Gets's e Set's)

   Método Adicional

   Mostra_Hospital()

2. No **programa principal** deve:

   a. **Criar as variáveis globais para:**

      i. **Criar um contador universal** para inserir registos;

      ii. **Criar uma lista ordenada vazia** para armazenar os nossos dados, onde o nome da lista deve ser a seguinte combinação: "lista_" + o nome da classe (ex: lista_produtos);

   b. O programa deve ter **um menu de escolha para que o utilizador possa optar pelas seguintes opções**:

- Adicionar
- Alterar
- Listar
- Eliminar
- Sair do Programa
- Todas as outras opções, dar a mensagem de erro "Opção inválida"

Este menu deve ser repetido tantas vezes até que o utilizador selecione a opção "Sair do programa", com uma estrutura de repetição: do … while.

Caso o utilizador **selecione** uma operação especifica (adicionar, alterar, listar e remover), **deve criar um procedimento e depois invocar o mesmo para cada caso do menu que foi criado.**

c. Caso seja selecionada a opção **Adicionar**, o programa deve pedir os dados ao utilizador e adicionar na lista ordenada;

d. Caso seja selecionada a opção **Alterar**, o programa deve:

i. **Pedir** qual o ID do registo a alterar;

ii. **Se** o valor a alterar existir:

- **Criar/exibir o novo menu** para questionar o que deseja alterar;
- **Para cada caso, utilizar o método de instância set da classe** para mudar o valor do campo do registo a alterar;

Senão, deve mostrar a mensagem de erro "Registo não encontrado" e voltar o menu principal;

e. Caso seja selecionada a opção **Listar**, o programa deve permitir percorrer e exibir os dados que existem na lista;

f. Caso seja selecionada a opção **Eliminar**, o programa deve fazer os seguintes passos:

i. **Pedir** qual o ID do registo a eliminar;

ii. **Se** o valor a eliminar existir:

- Usar a função **remove()** para remover o ID;

Senão, deve mostrar a mensagem de erro "Registo não encontrado" e voltar o menu principal;