UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO

PROGRAMAÇÃO AVANÇADA / 1ª unidade – 1ª avaliação
PROFESSOR: ADELARDO ADELINO DANTAS DE MEDEIROS

ALUNO: _____ MATRÍCULA: _____

Nesta avaliação, você deve desenvolver, em linguagem C++, uma versão bastante simplificada de um programa para armazenar e consultar voos de uma companhia aérea. Cada voo tem seu número, códigos dos aeroportos de origem e de destino e os horários de partida e de chegada. O programa deve ser construído da seguinte forma:

1. Defina uma `class` adequada para representar um voo, denominada `Voo`, com as seguintes informações (privadas):
   a. Número do voo (`unsigned int`: entre 1000 e 9999).
   b. Aeroporto de origem (`string` C++ de 3 letras: exemplo NAT = Natal).
   c. Aeroporto de destino (`string` C++ de 3 letras)
   d. Hora de partida (`unsigned int`: entre 0 e 2359; exemplo 1245 = 12h45min)
   e. Hora de chegada (`unsigned int`: entre 0 e 2359)

2. Preveja ao menos as seguintes funcionalidades para a classe `Voo`:
   a. Crie métodos de consulta para ter acesso em leitura aos dados privados.
   b. Sobrecarregue o `operator>>` para ler as coordenadas de um `Voo` a partir de valores fornecidos via teclado. Faça as críticas necessárias nos valores.
   c. Sobrecarregue o `operator<<` para imprimir um `Voo` em tela.

3. Utilizando o tipo `Voo`, defina uma `class`, denominada `ListaVoos`, adequada para representar a lista de voos de uma companhia. Uma lista de voos é composta por uma sequência de voos, armazenada em um array de `Voo`'s.

4. Preveja ao menos as seguintes funcionalidades para a classe `ListaVoos`:
   - Desenvolva o método `inserir`, para acrescentar um `Voo` a uma `ListaVoos`. O novo voo deve entrar no fim da lista, após os voos já existentes. Lembre-se que, para adicionar um novo voo, é necessário:
     a. Criar um ponteiro provisório.
     b. Alocar uma nova área de memória para esse ponteiro, com capacidade para mais um voo.
     c. Copiar os voos antigos para as primeiras posições da nova área.
     d. Colocar o novo voo como último elemento da nova área.
     e. Liberar a área de memória antiga.
     f. Fazer com que a `ListaVoos` aponte para a nova área de memória.
   - Programe uma função `imprimir` (sem parâmetros) para imprimir uma `ListaVoos`, utilizando várias chamadas ao `operator<<` da classe `Voo`. Deve ser impressa toda a lista de voos.
   - Programe uma função `imprimir` que receba como parâmetro um código de aeroporto (uma `string` C++), representando o aeroporto de origem. Devem ser impressos todos os voos da `ListaVoos` que partem desse aeroporto, utilizando várias chamadas ao `operator<<` da classe `Voo`.

5. Finalmente, faça um programa principal que utilize os métodos das classes programadas anteriormente para criar um menu com as seguintes opções para o usuário:
   - Inserir um novo voo via teclado.
   - Imprimir todos os voos cadastrados.
   - Imprimir todos os voos que partem de um determinado aeroporto, cujo código é digitado pelo usuário.

Apresenta-se a seguir um esboço do código em C++ que deve ser utilizado para iniciar o desenvolvimento. O programa principal (`main`) deverá ser exatamente igual ao apresentado, sem nenhuma modificação, acréscimo ou supressão.

```
#include <iostream>
#include <string>

using namespace std;

class Voo
{
  ...
};

class ListaVoos
{
  ...
};

...

int main(void)
{
  ListaVoos LV;
  Voo V;
  string codigo;
  int opcao;

  cout << "PROGRAMA PARA IMPLEMENTAR UMA LISTA DE VOOS\n";
  do
  {
    do
    {
      cout << "1 - Inserir um novo voo na lista\n";
      cout << "2 - Imprimir todos os voos da lista\n";
      cout << "3 - Procurar um voo a partir da origem\n";
      cout << "0 - Terminar o programa\n";
      cin >> opcao;
    } while (opcao<0 || opcao>3);
    switch (opcao)
    {
    case 1:
      cout << "Voo a ser inserido:\n";
      cin >> V;
      LV.inserir(V);
      break;
    case 2:
      LV.imprimir();
      break;
    case 3:
      cout << "Aeroporto de origem a pesquisar:\n";
      cin >> codigo;
      LV.imprimir(codigo);
      break;
    default:
      break;
    }
  } while (opcao!=0);
  return 0;
}
```