Universidade do Minho	Escola de Engenharia da Universidade do Minho **Mestrado Integrado em Eng. Electrónica Industrial e Computadores** # Complementos de Programação de Computadores	**2014/2015** **MIEEIC** **(1º Ano)** **2º Sem**
	Luís Paulo Reis	
	Aula Prática 3d: Exercícios de Introdução à Programação Orientada a Objectos	

Objectivos:

Esta Folha de Exercícios destina-se a:

• Compreender os conceitos de classes e encapsulamento, dados público e dados privados.

• Compreender os conceitos de operador e sobrecarga de operadores.

Os exercícios aqui propostos deverão ser realizados no mais simples ambiente de desenvolvimento possível para a linguagem C: editor de texto de programação ou editor DevC++ e ferramentas da GCC (GNU Compiler Collection) e afins.

Exercício 1

Pretende-se escrever um simulador de um radio-despertador em C++, denominado "RadioDespertador". O programa deve então conter uma classe denominada `CRadioDespertador` com a seguinte estrutura:

```
class CTIME {
      int h, m, s;
    };

class CRadioDespertador
{
  private:
    CTIME currTime;
    CTIME alarm;
    int alarmDuration;
    string stationAlarm;
    vector<string> stationsName;
    vector<double> stationsFrequency;

  public:
    int addStation (string name, const char * stationDatabase);
    int removeStation (string name);
    int removeStation (double freq);

    void setTime (int hr, int mn, int sg);
    void setAlarm (int hr, int mn, int sg, int duration, string station);
    int isRinging();
};
```

Execute as tarefas seguintes, tendo sempre o cuidado de testar no programa principal (função `main`), todas as funções criadas. O código fonte do programa deve ser escrito em ficheiros com o nome "main_RadioDespertador.cpp", "RadioDespertador.h" e "RadioDespertador.cpp".

a) Implemente o membro-função

```
int CRadioDespertador::addStation (string name, const char * stationDatabase);
```

que adiciona a estação com nome `name` à lista de estações pre-definidas. A frequência da estação é lida de um ficheiro `stationDatabase` com a informação de todas as estações conhecidas. O ficheiro está organizado em blocos de 2 linhas, contendo o nome da estação e a frequência:

```
Rádio Onda Viva
96.1
RFM
104.1
```

Se a estação não for encontrada no ficheiro ou já existir na lista de estações pre-definidas a função retorna -1; caso contrário retorna 0.

b) Implemente os membros-função

```
int CRadioDespertador::removeStation (string name);
int CRadioDespertador::removeStation (double freq);
```

Se a estação não for encontrada as funções retornam -1; caso contrário retornam 0;

c) Implemente o membro-função

```
void CRadioDespertador::setTime (int h, int m, int s);
void CRadioDespertador::setAlarm (int h, int m, int s, int duration, string
   station);
int CRadioDespertador::isRinging();
```

em que `duration` é a duração do alarme em minutos e `station` é a estação que deve tocar quando o alarme tocar.

A função `int isRinging();` retorna -1 se o alarme ainda não começou a tocar, 0 se se encontra a tocar e 1 se já tocou.

A função `void setAlarm(…);` verifica, no vector `stationsName`, se a estação escolhida está programada; caso não esteja deve adicioná-la à lista com o auxílio da função:

```
int CRadioDespertador::addStation (string name, const char *
   stationDatabase);
```

d) Implemente o operador ++ prefix e postfix, que incrementam em 1 segundo o tempo actual:

```
CRadioDespertador &operator++ ();
CRadioDespertador &operator++ (int);
```