```
/*
 * Calculateur_VRMS
 *
 * Une classe pour réaliser le calcul de la tension rms qui
 * représente le niveau sonore à l'entrée du capteur Electret
 * MAX4466.
 *
 * Voir les notes de cours "Conception des objets (IIB)" pour les
 * calculs à effectuer.
 *
 *
 * Note: La tension rms calculée est disponible en volt rms et
 *       en dBV.
 *
 * Convention:
 *  Variables -> camelCase
 *  Classes, fonctions -> PascalCase
 *  Constantes, types utilisateurs -> SNAKE_CASE
 *
 * GPA788 - ETS
 * T. Wong
 * 09-2018
 * 08-2020
 */
#ifndef CALCULATEUR_VRMS_H
#define CALCULATEUR_VRMS_H

class Calculateur_VRMS {
public:
  // Constructeur
  // Paramètres: aPin  - broche reliée au capteur sonore.
  //             vMax - tension max à la sortie du capteur sonore.
  //           maxADC - valeur max à la sortie de l'ADC du ucontrôleur
  Calculateur_VRMS(uint8_t aPin = A0, double vMax = 3.3, int16_t maxADC = 1024) {
    m_APin = aPin; m_VMax = vMax; m_AdcMax = maxADC;
    m_Amplitude =  m_Vrms = m_dBV = m_TmpVrms = 0.0;
    m_NbTotalSamples = m_NbSamples = 0;
    m_VDC_OFFSET = m_VMax / 2.0;
    m_C1 = m_VMax / m_AdcMax;
  }
  // Empêcher l'utilisation du constructeur de copie
  Calculateur_VRMS(const Calculateur_VRMS& other) = delete;
  // Empêcher l'utilisation de l'opérateur d'assignation
  Calculateur_VRMS& operator=(const Calculateur_VRMS& other) = delete;
  // Empêcher l'utilisation du constructeur par déplacement
  Calculateur_VRMS(Calculateur_VRMS&& other) = delete;
  // Empêcher l'utilisation de l'opérateur de déplacement
  Calculateur_VRMS& operator=(Calculateur_VRMS&& other) = delete;

  ~Calculateur_VRMS() {}                    // Destructeur

  /* ------------------------------------------------------------
     Accesseurs des données membres de la classe
     ------------------------------------------------------------ */
  inline uint16_t GetnbSamples() const { return m_NbSamples; }
  inline uint16_t GetTotalSamples() const { return m_NbTotalSamples; }
  inline double GetVrms() const { return m_Vrms; }
  inline int16_t GetAmplitude() const { return m_Amplitude; }
  inline double GetdBV() const { return m_dBV; }
```

```
60    inline uint8_t GetAPin() const { return m_APin; }
61    inline double GetVMax() const { return m_VMax; }
62    inline int16_t GetAdcMax() const { return m_AdcMax; }
63
64    inline void SetAPin(uint8_t A_Pin){ m_APin = A_Pin; }
65
66    /* -----------------------------------------------------------
67       Services publics offerts
68       ----------------------------------------------------------- */
69    // Accumuler les valeurs lues du capteur sonore dans le but de
70    // calculer la valeur rms du signal sonore.
71    // Note: La temporisation est la responsabilité de l'utilisateur.
72    void Accumulate() {
73
74       //1 seconde pour accumulate
75        m_Amplitude = analogRead(m_APin);
76        // Convertir en volts
77        double v = (m_Amplitude * m_C1) - m_VDC_OFFSET;
78        // Accumuler v^2
79        m_TmpVrms += (v * v);
80        ++m_NbSamples;
81
82    }
83
84    // Calculer la valeur rms du signal sonore.
85    // Note: La temporisation est la responsabilité de l'utilisateur.
86    void Compute() {
87      m_NbTotalSamples += m_NbSamples;
88      m_Vrms = sqrt(m_TmpVrms / m_NbSamples);
89      m_dBV = 20.0 * log10(m_Vrms);
90      // RAZ le décompte des échantillons
91      m_NbSamples = 0;
92      // RAZ le cumule des v^2
93      m_TmpVrms = 0.0;
94    }
95
96  private:
97    uint8_t m_APin;                          // Broche du Capteur sonore
98    double m_VMax;                           // VREF est à 3.3 V par défaut
99    int16_t m_AdcMax;                        // ADC à 10 bits
100   double m_VDC_OFFSET;                     // Tension décalage (niveau CC)
101   double m_C1;                             // Conversion bits -> volt
102   int16_t m_Amplitude;                     // Signal échantillonné en bits
103   double m_Vrms;                           // Valeur Vrms
104   double m_TmpVrms;                        // Pour acumuler v au carré
105   double m_dBV;                            // Valeur dBV
106   uint16_t m_NbTotalSamples;               // Nb. total des échantillons
107   uint16_t m_NbSamples;                    // Nb. d'échantillons
108 };
109
110 #endif
```