Etude de cas d'une classe « Acquisition »

Présentation

On souhaite acquérir une série de mesures de différence de potentiel - ddp.
On dispose d'une fonction *capteur()* qui retourne la valeur mesurée sous la forme d'un flottant. La fonction reçoit en paramètre la période d'échantillonnage (en ms, valeur entière). La période d'échantillonnage représente le temps qui doit s'écouler entre 2 mesures.
On met en œuvre une classe *Acquisition* modélisée ci-dessous.
Principe de l'acquisition d'une mesure :

Un nombre d'échantillons de la *dpp* sont prélevés et stockés dans *mesure[]*. La moyenne des mesures est effectuée et est stockée dans *resultat[]*. A la fin de l'acquisition de la série, le tableau *resulat* contient les valeurs de la série et le nombre d'éléments constituant la série est portée par *nbreResultats*.

Pour réaliser l'acquisition d'une série de mesures la procédure décrite ci-dessus est utilisée autant de fois qu'il y a de mesures dans la série.

Extrait d'un programme de mise en œuvre :

```
Acquisition ddp;
const int SERIE = 50;
const int ECHANTILLONS = 4;
float moyenne = 0;

//Acquérir une série de mesures
for(int serie=0; serie<SERIE; serie++) {
      for (int i=0; i<ECHANTILLONS; i++){
           ddp.ajouterMesure( capteur(100) );
      }
      ddp.calculerResultat();
}
```

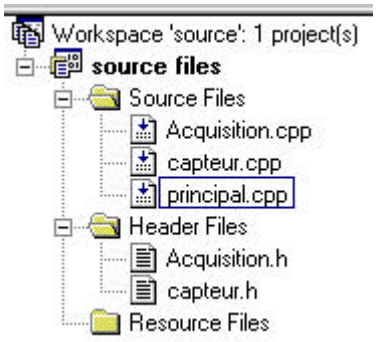
Modèle de la classe

Acquisition
- nbreResultats : int = 0 - nbreMesures : int = 0 - resultat[100] : float - mesure[10] : float
- moyenneMesure() : float + calculerResultat() : void + ajouterMesure(valeur : float) : void <<const>> + get_nbreMesures() : const float <<const>> + get_mesure(index : int) : const float + ~Acquisition() + Acquisition()

Dictionnaire

int nbreResultats	Nombre de résultats stockés dans le tableau resultat[]. Initialisé à 0
int nbreMesures	Nombre de mesures stockées dans le tableau mesure. Initialisé à 0
float resultat[100]	Tableau contenant le résultat des mesures de la série
float mesure[10]	Tableau contenant les mesures acquises
float moyenneMesure()	Calcule la moyenne des mesures stockées dans le tableau mesure[] et retourne le résultat
calculerResultat()	- Calcule la moyenne des mesures stockées dans mesure[] en appelant la méthode moyenneMesure(). - Ajoute le résultat au tableau resultat[] - Ajoute 1 à nbreResultats et met à 0 nbreMesures
ajouterMesure(float)	Ajoute une valeur au tableau mesure[] et ajoute 1 à nbreMesures
float get_nbreMesures()	Retour la valeur de nbreMesures
int get_mesure(int)	Retour la valeur stockée dans mesure[] dont l'index est donné en paramètre
Acquisition()	
~Acquisition()	

Sujet



Ci-contre est présenté une image décrivant le projet de développement :

1. Le développeur compile <u>tous les fichiers</u>. Donnez le nom et l'extension du ou des fichiers qui est (sont) produit(s).
2. Rappelez le rôle du constructeur de la classe *Acquisition.*

3. Qui peut accéder aux membres publics de la classe *Acquisition* ?
4. A partir du modèle de la classe et du dictionnaire, complétez la déclaration de la classe *Acquisition*

```
Acquisition();
~Acquisition();
const float get_mesure(int index) const;
const int get_nbreMesures() const;
void ajouterMesure(float valeur);
void calculerResultat();
```

```
nbreMesures
nbreResultats
```

5. Qui peut accéder à la fonction membre *calculerResultat()* ?
6. Déterminez la taille théorique d'une instance de la classe *Acquisition*.
7. Est-ce que *ddp* de l'extrait de programme page 1 aura la même taille théorique que celle de la question 6 ? Justifiez votre réponse.
8. Complétez le corps des fonctions :

```
const float Acquisition::get_mesure(int index) const {
}

const int Acquisition::get_nbreMesures() const {
}

Acquisition::Acquisition() {
}

float Acquisition::moyenneMesure() {
}
```

9. A partir de l'analyse de l'extrait de programme page1, déterminez le temps nécessaire pour constituer la série de mesures de *ddp*.
10. Etablissez le prototype de la fonction *capteur()* utilisez dans l'extrait de programme page 1.

11. Enrichir de l'extrait de programme page 1.

On souhaite après avoir constitué la série de mesures de *ddp* rechercher la valeur maximum. Donnez le fragment de code qui viendrait s'insérer derrière l'extrait de programme permettant de réaliser cette opération.