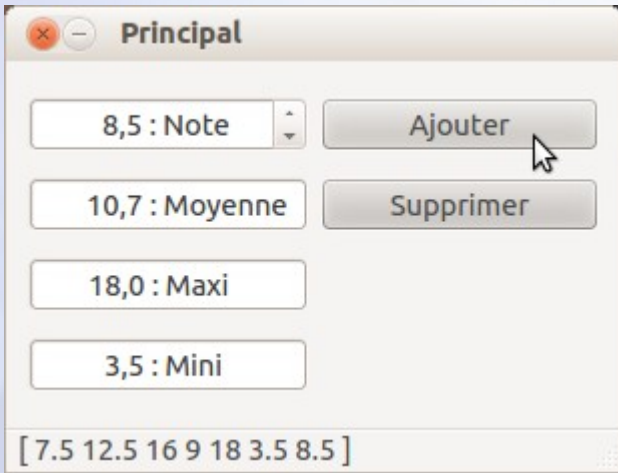


Ce projet permet de valider le cours concernant l'utilisation de fonctions afin de réduire la complexité au travers de la mise en œuvre du projet sur la gestion des notes d'un élève.



x PRINCIPAL.UI

The design view shows the UI layout with spin boxes for 'Note', 'Moyenne', 'Maxi', and 'Mini', and buttons for 'Ajouter' and 'Supprimer'. The Object Inspector lists the following objects and their classes:

Objet	Classe
Principal	QMainWindow
centralWidget	QWidget
boutonAjouter	QPushButton
boutonSupprimer	QPushButton
maxi	QDoubleSpinBox
mini	QDoubleSpinBox
moyenne	QDoubleSpinBox
note	QDoubleSpinBox
barreEtat	QStatusBar

The Signal/Slot editor shows the following connections:

Émetteur	Signal	Receveur	Slot
boutonAjouter	clicked()	Principal	ajouter()
boutonSupprimer	clicked()	Principal	supprimer()

The Properties panel for 'moyenne : QDoubleSpinBox' shows the following settings:

Propriété	Valeur
prefix	
suffix	: Moyenne
decimals	1
minimum	0.000000
maximum	20.000000
singleStep	1.000000
value	0.000000

At the bottom, the taskbar shows: 1 Problèmes, 2 Résultat de la recherche, 3 Sortie de l'application, 4 Sortie de compilation.



```
x PRINCIPAL.H
```

```
#ifndef PRINCIPAL_H
#define PRINCIPAL_H
```

```
#include <QMainWindow>
#include "ui_principal.h"
```

```
class Principal : public QMainWindow, public Ui::Principal
{
    Q_OBJECT
```

```
public:
    explicit Principal(QWidget *parent = 0);
```

```
private slots:
    void ajouter();
    void supprimer();
};
```

```
#endif // PRINCIPAL_H
```

```
x PRINCIPAL.CPP
```

```
#include "principal.h"
```

```
const unsigned NombreMax = 10;
double notes[NombreMax];
unsigned nombre = 0;
```

```
void ajouterNote(double note)
{
    if (nombre < NombreMax) notes[nombre++] = note;
}
```

```
void supprimerDerniereNote()
{
    if (nombre > 0) nombre--;
}
```

```
QString visuTableau()
{
    if (nombre == 0) return "[ ]";
    QString chaine = "[ ";
    for (unsigned i = 0; i < nombre; i++)          chaine.append(QString::number(notes[i])).append(' ');
    return chaine.append("]");
}
```

```
double calculMoyenne()
{
    if (nombre == 0) return 0.0;
    double somme = 0.0;
    for (unsigned i = 0; i < nombre; i++) somme += notes[i];
    return somme / nombre;
}
```

```
double plusGrande()
{
    if (nombre == 0) return 0.0;
    double plusHaute = notes[0];
    for (unsigned i = 0; i < nombre; i++)
        if (notes[i] > plusHaute) plusHaute = notes[i];
    return plusHaute;
}
```

```
double plusPetite()
{
    if (nombre == 0) return 0.0;
    double plusPetite = notes[0];
    for (unsigned i = 0; i < nombre; i++)
        if (notes[i] < plusPetite) plusPetite = notes[i];
    return plusPetite;
}
```





```
Principal::Principal(QWidget *parent) : QMainWindow(parent)
```

```
{
    setupUi(this);
}

void Principal::ajouter()
{
    ajouterNote(note->value());
    moyenne->setValue(calculMoyenne());
    maxi->setValue(plusGrande());
    mini->setValue(plusPetite());
    barreEtat->showMessage(visuTableau());
}

void Principal::supprimer()
{
    supprimerDerniereNote();
    moyenne->setValue(calculMoyenne());
    maxi->setValue(plusGrande());
    mini->setValue(plusPetite());
    barreEtat->showMessage(visuTableau());
}
```

x DÉCLARATION DES FONCTIONS

Dans le codage précédent, nous avons placé la définition des fonctions de calcul avant leurs utilisations par les méthodes de rappel propres aux slots.

Il est quelque fois plus judicieux de placer en premier lieu les méthodes principales avant les fonctions annexes qui ne servent qu'aux calculs de base. Dans ce cas là, il est nécessaire de déclarer ces fonctions subalternes dans le fichier en-tête.

/home/manu/Dropbox/Projets/notes/principal.h

```
#ifndef PRINCIPAL_H
#define PRINCIPAL_H

#include <QMainWindow>
#include "ui_principal.h"

class Principal : public QMainWindow, public Ui::Principal
{
    Q_OBJECT

public:
    explicit Principal(QWidget *parent = 0);
private slots:
    void ajouter();
    void supprimer();
};

void ajouterNote(double note);
void supprimerDerniereNote();
QString visuTableau();
double calculMoyenne();
double plusGrande();
double plusPetite();

#endif // PRINCIPAL_H
```

