4ETI IMI, Examen [C++]
Programmation générique en C++
CPE Lyon

2014-2015 (1ere session)
durée 2h

Tous documents autorisés. Calculatrices interdites.

Répondez aux questions sur une copie séparée

Le sujet comporte 4 pages

Le barème approximatif est indiqué pour les grandes parties. Notez que le barème est donné à titre purement indicatif et pourra être adapté par la suite.

En cas de doute sur la compréhension de l'énoncé, explicitez ce que vous comprenez et poursuivez l'exercice dans cette logique.

Note préliminaire:

- Toutes les questions concernent le language de programmation C++ dans le cadre du developpement de logiciels sur architecture PC standard.
- Nous vous invitons à commenter le code et les réponses. En particulier, en cas d'ambiguité de compréhension d'une question, ajoutez toutes remarques et illustrations supplémentaires permettant d'expliquer votre démarche.
- Sauf mention contraire explicite, on supposera que l'on dispose d'un système Linux standard fonctionnant correctement sur un PC récent 32 ou 64 bits (identiques aux conditions de TP des PC de CPE: *int* étant encodé sur 4 octets, pointeurs étant encodés sur 8 octets en 64 bits).
- On supposera que le code C++ est compilé avec une version récente de g++ sous la norme C++11 (ou ultérieure) identique aux conditions de TP des PC de CPE (avec l'option -std=c++11).
- On supposera dans chaque cas que les `#include` des en-tête standards nécessaires à la bonne compilation et exécution des programmes décrits sont correctement installés et appelés (ex. iostream, string, etc).

1 Qt

(5 points)

On considère l'interface graphique développée en Qt à l'aide de QtDesigner telle qu'illustrée en figure 1. De manière identique à votre TP, chaque widget Qt est placé dans une classe conteneur `ui`, et est désigné par une variable du même nom que le texte qui est affiché dans l'interface.

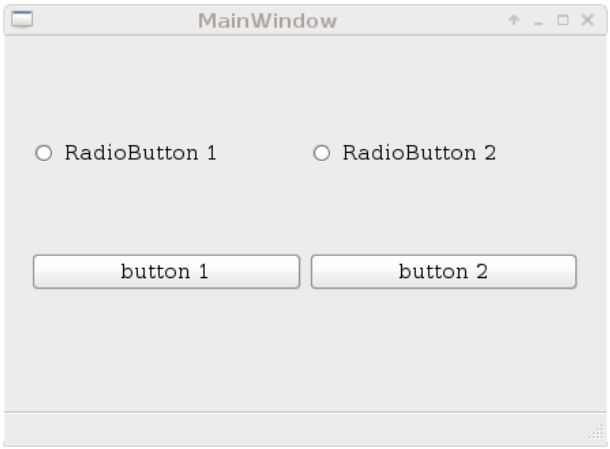

Figure 1: Interface graphique proposée.

Similairement au cas de votre TP, le code de gestion des actions de ces Widgets est écrit dans une classe `myWindow`. Le fichier `myWindow.hpp` contient le code suivant:

```
#pragma once

#include <QMainWindow>
namespace Ui{class MainWindow;}

class myWindow: public QMainWindow
{
    Q_OBJECT
public:
    myWindow(QWidget *parent=nullptr);
private slots:
    void actionRadioButton();
    void actionButton1();
    void actionButton2();
private:
    Ui::MainWindow *ui;
};
```

Le fichier `myWindow.cpp` contient le code suivant:

```
#include "myWindow.hpp"
#include "ui_mainwindow.h"

#include <iostream>
#include <QtGui>

myWindow::myWindow(QWidget *parent)
    :QMainWindow(parent),ui(new Ui::MainWindow)
{
    ui->setupUi(this);
```

```
    connect(ui->Button_1,SIGNAL(clicked()),this,
                            SLOT(actionButton1()));
    connect(ui->Button_2,SIGNAL(clicked()),this,
                            SLOT(actionButton2()));
    connect(ui->radioButton_1,SIGNAL(clicked()),this,
                            SLOT(actionRadioButton()));
    connect(ui->radioButton_2,SIGNAL(clicked()),this,
                            SLOT(actionRadioButton()));
}

void myWindow::actionRadioButton()
{
    std::cout<<ui->radioButton_1->isChecked()
             <<" ; "<<ui->radioButton_2->isChecked()
             <<std::endl;
}
void myWindow::actionButton1()
{
    actionRadioButton();
}
void myWindow::actionButton2()
{
    std::cout<<"Button 2 clicked"<<std::endl;
}
```

On clique respectivement sur le *RadioButton 1*, puis sur *RadioButton 2*, puis *button 1*, et enfin sur *button 2*.

Question 1 *Décrivez ce qu'affiche ce programme sur la ligne de commande en expliquant votre raisonnement.*

2 Héritage et polymorphisme

(5 points)

Soit le code suivant:

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

struct Base
{
    Base(){}
    virtual ~Base(){}

    virtual std::string f1() const {return "Base";}
    std::string f2() const {return "Base";}
    std::string f3() const {return "Base";}
};

struct Derive : Base
{
    Derive():Base(){}
    virtual ~Derive(){}

    std::string f1() const {return "Derive";}
    virtual std::string f2() const {return "Derive";}
    std::string f3() const {return "Derive";}
};
```

```
template <typename T> void action_0(T const& x)
{
    std::cout<<x.f1()<<","<<x.f2()<<","<<x.f3()<<std::endl;
}
template <typename T> void action_1(T x) {action_0(x);}
void action_2(Base const& x) {action_0(x);}
void action_3(Base& x) {action_0(x);}
void action_4(Base x) {action_0(x);}

int main()
{
    Derive d;
    action_0(d); action_1(d);
    action_2(d); action_3(d);
    action_4(d);
    return 0;
}
```

Question 2 *Ecrivez ce qu'affiche l'exécution de ce programme en expliquant votre raisonnement.*

3 Vocabulaire

(5 points)

Question 3 *Définissez précisément en quelques lignes à quoi correspond les termes suivants.*

- Qt
- moc
- uic
- template
- CMakeLists.txt

4 Généricité

(5 points)

On vous demande de définir une fonction `multiplication_matrice_vecteur` qui vient multiplier une matrice quelconque avec un vecteur quelconque lorsque cela est possible. Votre fonction sera générique au sens où elle acceptera tous les types possibles de matrices et vecteurs possédants les propriétés attendues.

Question 4 *Ecrivez l'en-tête de la fonction correspondant à cette demande.*

Question 5 *Décrivez quelles fonctionnalitées devront posséder les classes de matrices et vecteurs passés en arguments pour que votre fonction puisse être appliquée (plusieurs solutions valides possibles).*

Question 6 *Écrivez l'implémentation complète de votre fonction de multiplication matricielle.*