HOCHSCHULE RUHR WEST
UNIVERSITY OF APPLIED SCIENCES
INSTITUT INFORMATIK

25.03.2025

Klausur

MMI und GUI-Programmierung

Wintersemester 2024/25

Studiengänge: AI, MTI, WI

Prüfer: Prof. Dr. Malte Weiß

Persönliche Angaben

Nachname	Vorname	Matrikelnummer

Bewertung

Frage	Punkte	Erreicht
1	8	
2	9	
3	8	
4	8	
5	12	
6	12	

Frage	Punkte	Erreicht
7	3	
8	26	
9	8	
10	6	
Gesamt:	100	

Ablauf der Prüfung

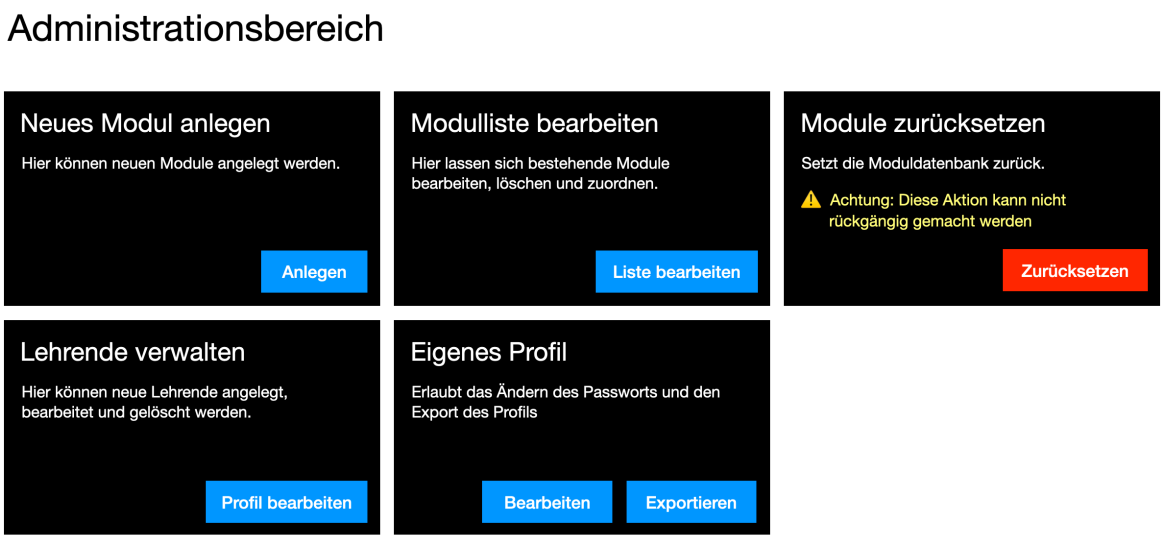
- **Zeit für die Lösung dieses Aufgabenblatts**: 120 Minuten.
- **Erlaubte Hilfsmittel**: Cheat Sheet am Ende der Klausur. Handgeschriebener, ein- oder beidseitig beschriebener Notizzettel im DIN-A4-Format.
- **Nicht erlaubte Hilfsmittel**: Mobiltelefon, Kommunikation (E-Mail, Web, SMS, etc.). Die Verwendung eines nicht erlaubten Hilfsmittels wird als Täuschungsversuch gewertet.
- **Stift und Papier**: Schreiben Sie ausschließlich auf dem Papier der Aufgabenblätter. Sollten die Zwischenräume nicht ausreichen, können Sie zusätzliches Papier von der Aufsicht erhalten. Beschriften Sie dieses sofort mit ihrem Namen und Ihrer Matrikelnummer. Eigenes Papier ist nicht zulässig. Schreiben Sie ausschließlich mit dokumentenechten schwarzen oder blauen Stiften.
- **Vollständigkeit des Aufgabenblatts**: Die Klausur besteht aus den Aufgabenseiten 5 – 23 und einer Funktionsreferenz auf Seite R1. Die Aufgabenblätter sind doppelseitig bedruckt. Prüfen Sie, ob alle Blätter vorhanden sind.
- **Fragen**: Melden Sie sich, wenn Sie eine Frage haben. Die Aufsicht kommt zu Ihnen, verlassen Sie nicht unaufgefordert Ihren Platz.
- **Zwischenschritte**: Geben Sie Zwischenschritte an. So können Sie selbst bei falschem Endergebnis einen Teil der Punkte erreichen.

Mensch-Maschine-Interaktion

1. (a) Was ist ein Wireframe und zu welchem Zweck wird es erstellt? (4)

(b) Alice und Bob wollen ein Unternehmen mit folgendem Ziel gründen: "Wir möchten eine App für alte Menschen entwickeln, damit sie sich besser mit digitalen Produkten zurecht zu finden." Wieso sind "alte Menschen" eine ungünstig definierte Zielgruppe für die Entwicklung von Nutzerschnittstellen? (4)

2. Betrachten Sie die folgende GUI einer Web-Anwendung: (9)



Nennen Sie **drei** Gestaltgesetze, die hier eingesetzt werden, und beschreiben Sie, welchen Effekt sie bei der Wahrnehmung der Anwendung haben.

(Bitte lösen Sie die Aufgabe auf der nächsten Seite.)

3. (a) Wieso sollte man aus Nutzersicht eine Undo-Funktion[1] zur Verfügung stehen? (4)

(b) Eine wichtige Regel der Usablity lautet "Sei konsistent und vorhersehbar". Wie berücksichtigt man diese Regel bei der Entwicklung einer Smartphone-App? (4)

[1] Eine Undo-Funktion erlaubt es, Nutzeraktionen rückgängig zu machen.

C++

4. (a) Nennen Sie zwei Regeln, die eingehalten werden müssen, wenn man Operatoren in C++ überlädt. (4)

(b) Welche Auswirkung hat das Schlüsselwort `friend` in C++? (4)

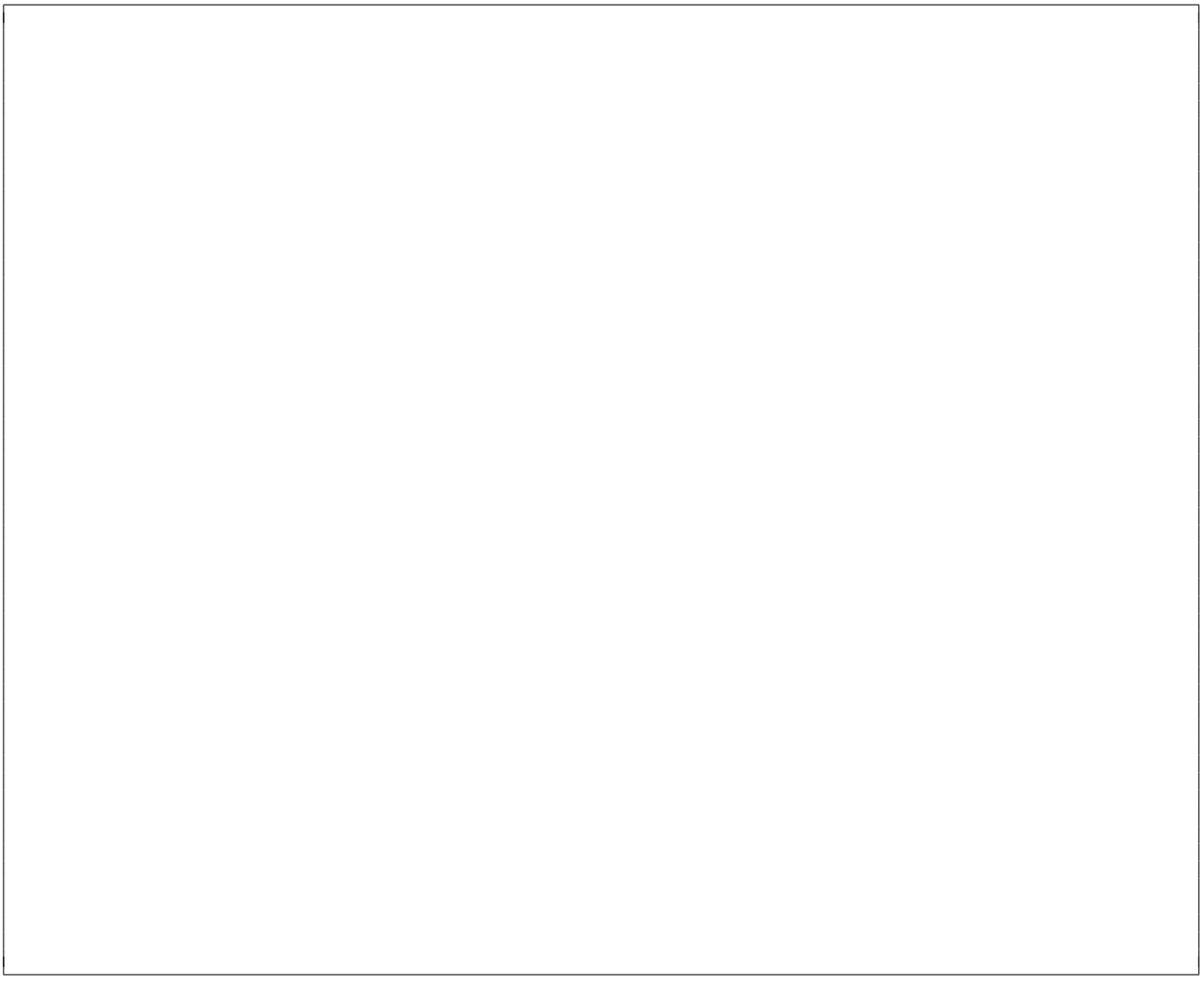
Polymorphie

5. Implementieren Sie die folgende Klassenhierarchie. (12)

   - Die Klasse `Person` stellt eine Person dar. Eine Person besitzt einen Namen (String).
   - Die Klasse `Student` stellt einen Studenten dar. Ein Student ist eine Person und besitzt ein zusätzliches Attribut für die Matrikelnummer.

   Implementieren Sie beide Klassen. Jede Klasse sollte einen Konstruktor besitzen, der alle Attribute setzt, und Getter für jedes Attribut. Zusätzlich soll es eine Methode `toString()` geben. Sie liefert eine String-Repräsentation eines Objekts zurück, die alle Attribute beschreibt.

   Tipp: Für die `toString()`-Methode können Sie einen String-Stream (sstream) einsetzen, um mehrere Attribut-Ausgaben über den `<<`-Operator zusammenzusetzen.



Operatoren

6. Die Klasse `Set` soll eine einfache Klasse darstellen, die Zahlen speichert. Jede Zahl darf dabei nur ein einziges Mal in der Menge vorkommen. (12)

   Implementieren Sie nun die folgenden Operatoren:

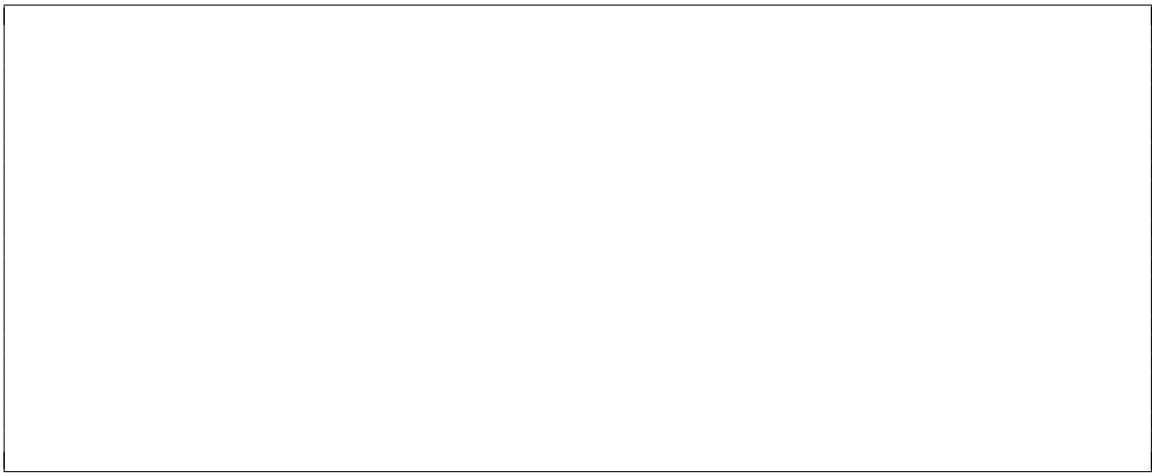
   - Der `<<` Operator soll es ermöglichen, eine Zahl der Menge hinzuzufügen. Das geschieht aber nur, wenn die Zahl noch nicht in der Menge vorkommt. Ansonsten hat der Operator keine Wirkung.
   - Der `|` Operator vereinigt zwei Mengen.
   - Der `|=` Operator vereinigt die aktuelle Menge mit einer zweiten gegebenen Menge.

   Der folgende Code demonstriert den Einsatz der Klasse:

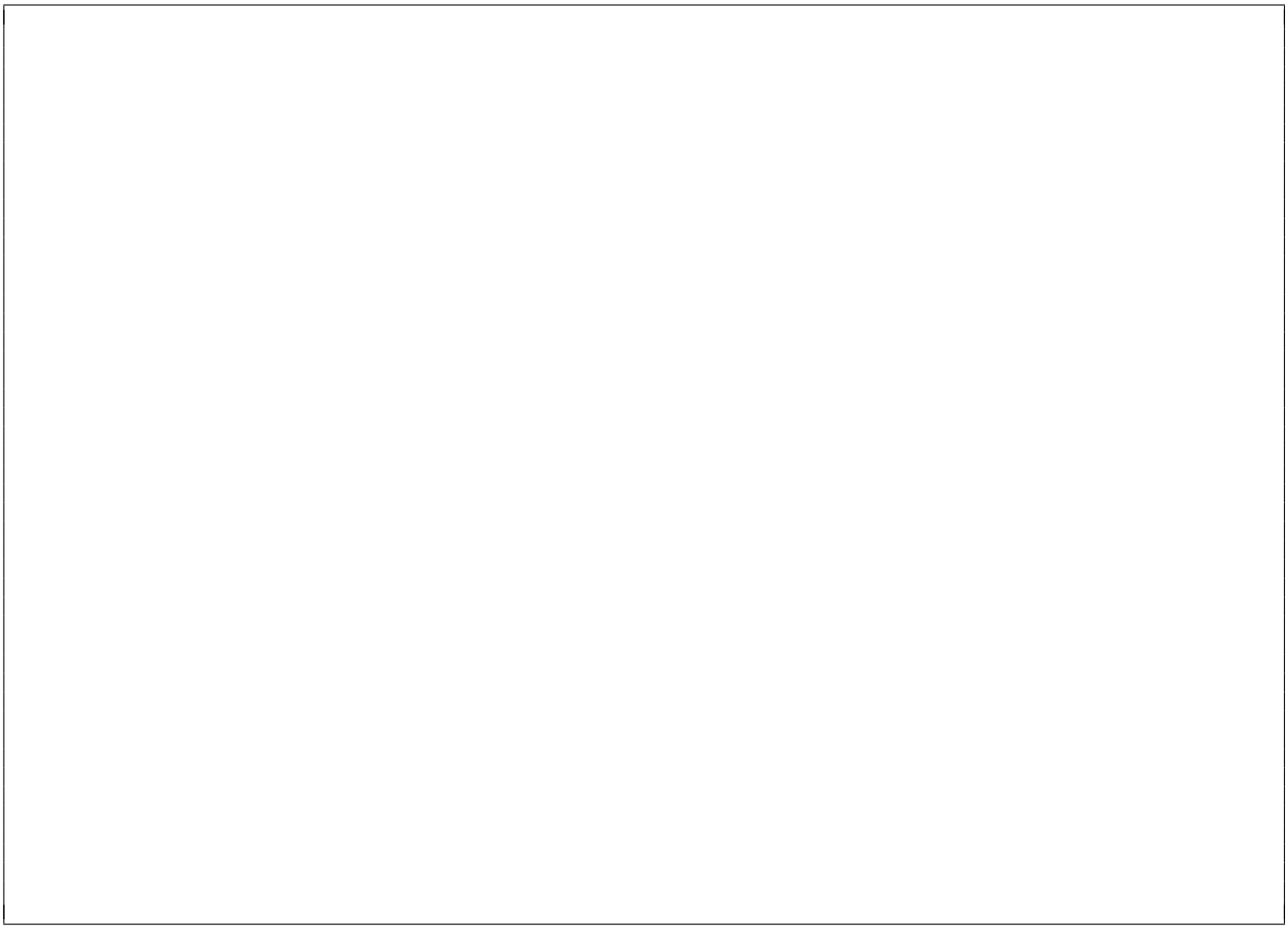
```
int main() {
    Set set1;
    set1 << 1 << 3 << 5;      // set1 = {1, 3, 5}
    set1 << 5 << 6;           // set1 = {1, 3, 5, 6}

    Set set2;
    set2 << 5 << 6 << 7;      // set2 = {5, 6, 7}

    Set set3 = set1 | set2;   // set3 = {1, 3, 5, 6, 7}

    set1 |= set2;             // set1 = {1, 3, 5, 6, 7}

    return 0;
}
```

   Ergänzen Sie den Code auf den nächsten Seiten.

```
class Set {
public:
    Set() {}
```



```
private:
    std::vector<int> m_values;
};
```

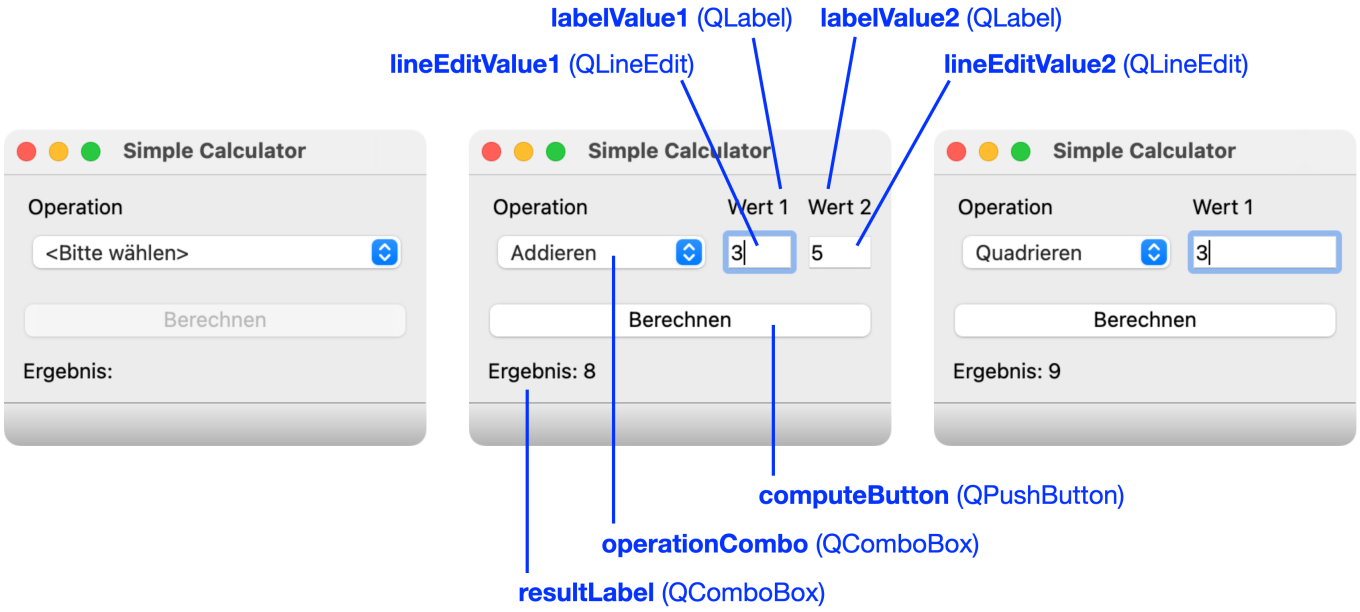


Qt

7. Die Qt-Klasse QObject unterstützt eine effiziente Speicherverwaltung. Obwohl in einer Qt-GUI sehr viele Objekte dynamisch (mit `new`) angelegt werden, reicht meist ein einziges `delete`, um alle Objekte wieder freizugeben. Beschreiben Sie den Mechanismus, der das sicherstellt. (3)

Qt – GUI

8. Betrachten Sie das folgende Wireframe einer GUI, mit der sich einfache Berechnungen durchführen lassen.



Die GUI wurde bereits im Qt-Designer umgesetzt. Implementieren Sie in der Hauptfensterklasse `MainWindow` die folgende Anwendungslogik:

- Ein Nutzer kann die Rechenoperation in einer Combo-Box `operationCombo` ausgewählen. In der Combo-Box gibt es drei Einträge: "<Bitte wählen>" (Index: 0), "Addieren" (Index: 1), "Quadrieren" (Index: 2)
- Wert 1 (Label und Eingabefeld) ist nur **sichtbar**, wenn entweder Addieren oder Quadrieren ausgewählt ist.
- Wert 2 (Label und Eingabefeld) ist nur **sichtbar**, wenn Quadrieren als Funktion ausgewählt ist.
- Der Knopf "Berechnen" ist nur **aktiviert**, wenn entweder Addieren oder Quadrieren ausgewählt ist und alle sichtbaren Werte-Felder nicht leer sind.
- Bei Klick auf "Berechnen" wird das Ergebnis der Berechnung angezeigt.

Tipps:

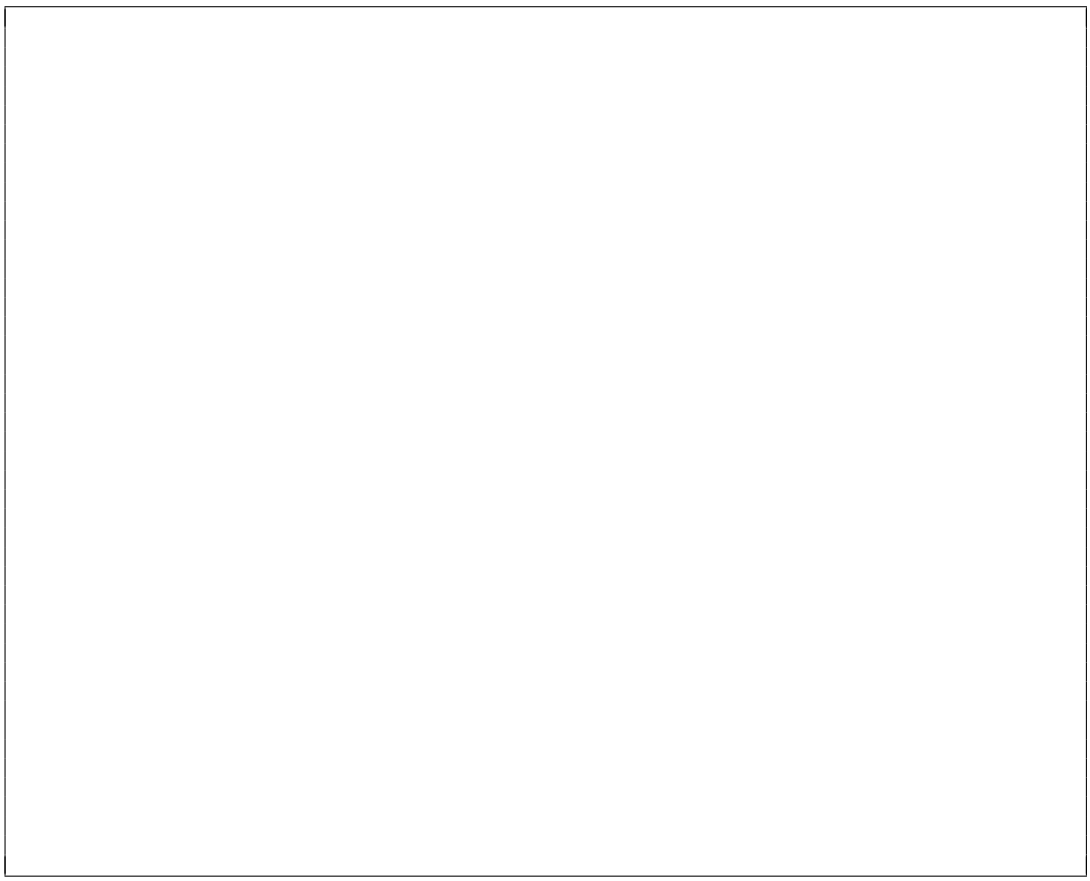
- Es bietet sich an, eine zentrale Methode zu schreiben, die die GUI-Dymnamik umsetzt, z.B. das (De)Aktivieren oder (Un)sichtbar-Machen von Elementen.
- Die Referenz im Anhang enthält eine Übersicht über sinnvolle Funktionen, insbesondere für die Combo-Box.

(a) Beschreiben Sie, welche Layout-Manager und Spacer eingesetzt werden, um die gezeigte GUI umzusetzen. Beachten Sie dabei, dass die Fenstergröße geändert werden kann. Sie dürfen die Aufgabe über eine Grafik, einen Text oder beides lösen. (6)

(b) Vervollständigen Sie den Code auf den folgenden Seiten, um die beschriebene Logik zu implementieren. Trennen Sie den Quelltext in Header- und Source-Datei sinnvoll auf. (20)

Headerdatei mainwindow.h

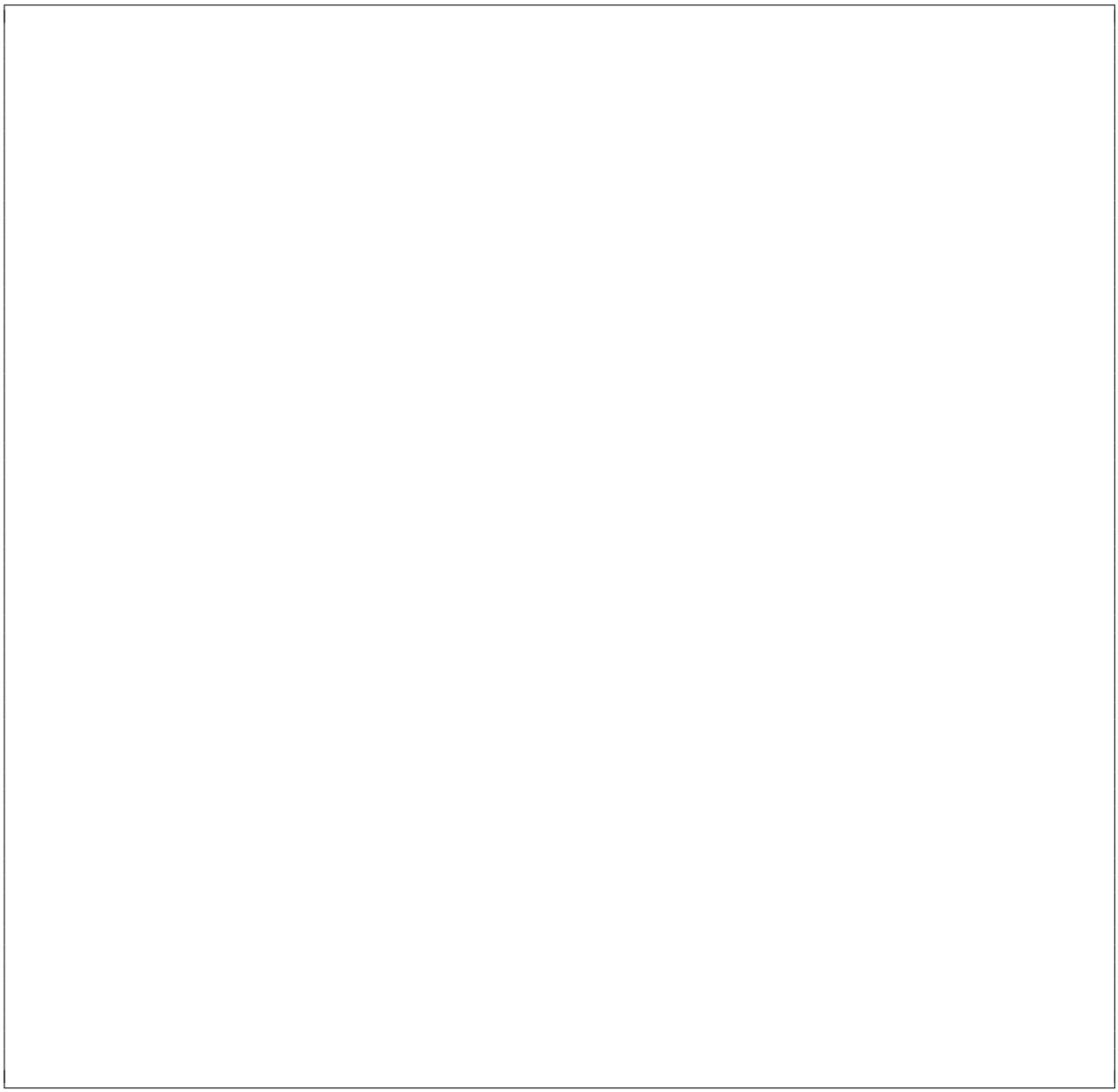
```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

class MainWindow : public QMainWindow {
   Q_OBJECT

public:
   MainWindow(QWidget *parent = nullptr);

private:
   Ui::MainWindow *ui;
```



```
};

#endif // MAINWINDOW_H
```

Sourcedatei mainwindow.cpp (Konstruktor)

```
#include "mainwindow.h"
#include <QMessageBox>

MainWindow::MainWindow(QWidget * parent): QMainWindow(parent) {
```



```
}

MainWindow::~MainWindow()
{
    delete ui;
}
```

Sourcedatei mainwindow.cpp (weitere Methoden)

Nebenläufigkeit

9. (a) Die preisgekrönte Qt-App “Grab the money” bietet die Möglichkeit auf Knopfdruck, Aktienkurse vorherzusehen. Ein Nutzer bemängelt, dass die App beim Drücken auf “Kurs berechnen” nicht mehr reagiert. Beim Klick auf den Knopf zeigt das Betriebssystem nach einiger Zeit sogar “App reagiert nicht” an und bietet an, die App zu terminieren. Erst wenn die Berechnung fertiggestellt ist, kann man wieder mit der GUI arbeiten. (4)

   Wieso reagiert die App nach dem Knopfdruck nicht mehr und was hätte der Entwickler besser machen können? Begründen Sie Ihre Antwort.

(b) Zeigen Sie anhand eines einfachen Codebeispiels, wie man verhindert, dass zwei Threads parallel den selben Code ausführen. (4)

Analyse

10. Was ist an den folgenden Codestellen falsch? Begründen Sie Ihre Antwort.

    (a) Die Klasse `MagicNumber` soll eine Zahl speichern, aber der Code kompiliert nicht: (3)

```
class MagicNumber {
public:
    MagicNumber(int number) {
        m_number = number;
    }
    int getNumber() const {
        return m_number;
    }

private:
    const int m_number;
};
```

(b) Die Methode `onSelectModule` soll einen nicht-modalen Qt-Dialog aufrufen. Der Dialog wird aber gar nicht angezeigt. Woran liegt das? (3)

```
void MainWindow::onSelectModule() {
    ModuleSelectionDialog dialog;
    dialog.show();
}
```

Qt Cheat Sheet

Signale

QPushButton / QAbstractButton

void clicked(bool checked = false)
void toggled(bool checked)
void pressed()
void released()

QComboBox

void currentIndexChanged(int index)

Methoden

QLabel

Text auslesen
QString text() const

Wert setzen
void setText(const QString &)
void setNum(int)

QComboBox

Index des ausgewählten Eintrags auslesen
int currentIndex() const

Eintrag mit Index auswählen
void setCurrentIndex(int)

QString

String mit Zahl erstellen
QString::number(int)
String in Zahl umwandeln
toInt()
toFloat()

Slots

QWidget

bool close()
void hide()
void setDisabled(bool disable)
void setEnabled(bool)
void setFocus()
void setHidden(bool hidden)
void setVisible(bool visible)
void setWindowModified(bool)
void setWindowTitle(const QString &)
void show()
void update()

QAbstractButton

void animateClick(int msec = 100)
void click()
void setChecked(bool)
void setIconSize(const QSize &size)
void toggle()

QPushButton *(erbt von QAbstractButton)*

void showMenu()

Layout-Manager

QHBoxLayout
QVBoxLayout
QFormLayout
QBoxLayout
QGridLayout
QStackedLayout

Thread-Synchronisation

QMutex
QReadWriteLock
QSemaphore