HOCHSCHULE RUHR WEST
UNIVERSITY OF APPLIED SCIENCES
INSTITUT INFORMATIK

20.09.2023

Klausur

MMI und GUI-Programmierung

Sommersemester 2023

Studiengänge: AI, MTI, WI

Prüfer: Prof. Dr. Malte Weiß

Persönliche Angaben

Nachname	Vorname	Matrikelnummer

Bewertung

Frage	Punkte	Erreicht
1	4	
2	4	
3	5	
4	8	
5	12	
6	16	

Frage	Punkte	Erreicht
7	6	
8	10	
9	20	
10	5	
11	10	
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 – 21 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.

Programmierung

1. Welche Auswirkung hat das Schlüsselwort `const` in C++, wenn es an einen Funktions-Member angehängt wird? (4)

   Beispiel:

   ```
   void Person::getName(int element) const {
      ...
   }
   ```

2. Beschreiben Sie wesentliche Eigenschaften von Klassen, die von QObject erben. (4)

3. Führt man in Qt lange Rechenoperationen durch, ohne Threads einzusetzen, kann der GUI-Thread blockieren. Was bedeutet das und was ist die Ursache für dieses Problem? (5)

Usability

4. Betrachten Sie das folgende Bild einer Kaffeemaschine: (8)

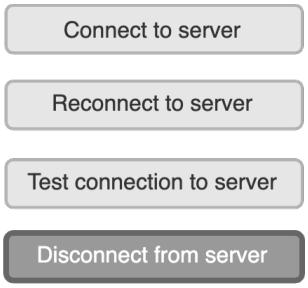
Welche Probleme besitzt das Bedienfeld (rechtes Bild) aus Usability-Sicht. Begründen Sie Ihre Antwort mit den Konzepten, die Sie in der Vorlesung kennengelernt haben.

Beantworten Sie die Frage auf der nächsten Seite.

Gestaltgesetze

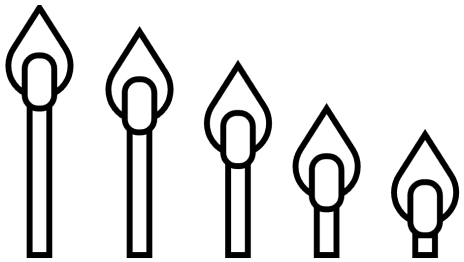
5. Betrachten Sie die folgenden Strukturen. Welches Gestaltgesetz wird hier gezielt umgesetzt oder verletzt? Nennen Sie das umgesetzte oder verletzte Gestaltgesetz *und* begründen Sie kurz ihre Entscheidung.

(a) (4)



(b) (4)

(c) (4)



C++ – Operatoren

6. Betrachten Sie die folgenden Struktur, die einen 2D-Vektor repräsentiert: (16)

```
struct Vec2 {
    float x, y;
};
```

Implementieren Sie die folgenden Operatoren für 2D-Vektoren in C++:

- Addition von zwei Vektoren (a + b). (4 P)

  Beispiel: $\begin{pmatrix}1\\2\end{pmatrix} + \begin{pmatrix}3\\-1\end{pmatrix} = \begin{pmatrix}4\\1\end{pmatrix}$

- Unärer Operator (-a). (4 P)

  Beispiel: $-\begin{pmatrix}1\\2\end{pmatrix} = \begin{pmatrix}-1\\-2\end{pmatrix}$

- Skalarprodukt (a * b). (5 P)

  Beispiel: $\begin{pmatrix}1\\2\end{pmatrix} * \begin{pmatrix}3\\-1\end{pmatrix} = 1*3+2*(-1) = 3-2 = 1$

- Operator, um einen Vector über `std::cout` auszugeben. (5 P)

Der folgende Code demonstriert die Nutzung der Operatoren:

```
int main2() {
    Vec2 u = {1, 2};
    Vec2 v = {3, 4};

    std::cout << u << endl;         // (1, 2)
    std::cout << v << endl;         // (3, 4)
    std::cout << (u + v) << endl;   // (4, 6)
    std::cout << -u << endl;        // (-1, -2)

    return 0;
}
```

Verwenden Sie dazu bitte die nächste Seite.

C++ – Virtuelle Methoden

7. Betrachten Sie den folgenden C++-Code: (6)

```
class TopClass {
public:
    void print() {
        printf("Top class 1\n");
    }
    virtual void printVirtual() {
        printf("Top class 2\n");
    }
};

class BottomClass : public TopClass {
public:
    void print() {
        printf("Bottom class 1\n");
    }
    virtual void printVirtual() {
        printf("Bottom class 2\n");
    }
};

int main() {
    TopClass* top = new BottomClass();
    top->printVirtual();
    top->print();

    return 0;
}
```

Was gibt der Code aus? Begründen Sie Ihre Antwort.

(Aufgabe bitte auf der nächsten Seite bearbeiten)

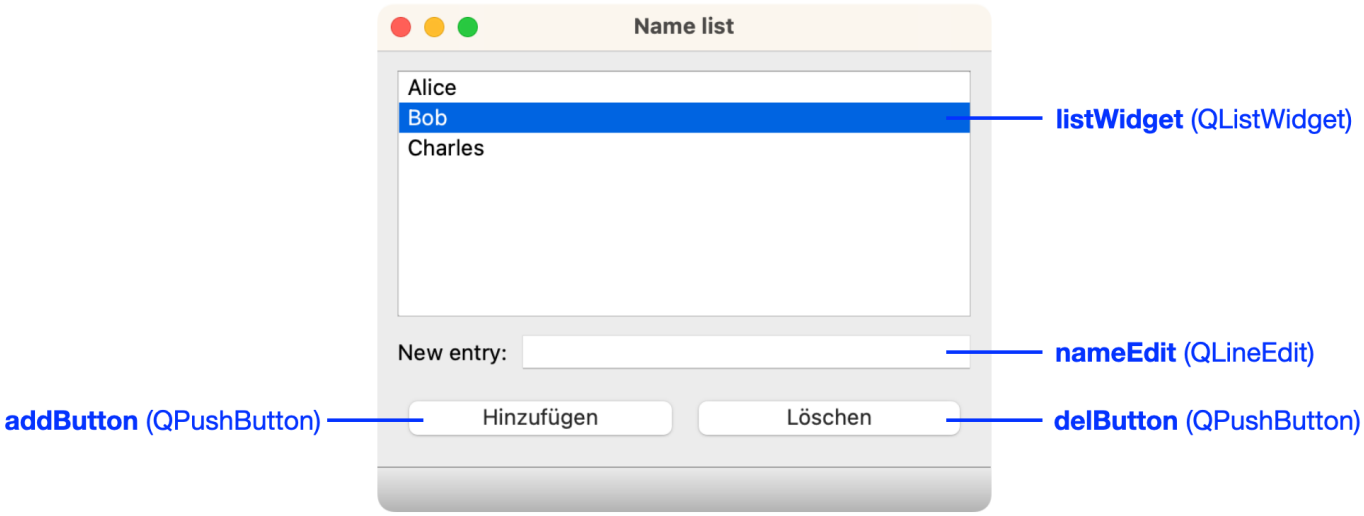
C++ – Generics

8. Schreiben Sie eine generische Klasse `Node`. Sie besitzt einen generischen Typ-Parameter T und repräsentiert einen Knoten in einer verketteten Liste. Die Klasse besitzt folgendes: (10)

   - Ein privates Attribut `content`, das den Inhalt des Knotens repräsentiert. Der Typ des Attributs sollte generisch sein.
   - Einen privaten Zeiger `node` auf das nächste Element.
   - Einen Standard-Konstruktor, der den Zeiger `node` auf `nullptr` setzt.
   - Getter und Setter.

Qt – GUI

9. Betrachten Sie das folgende Wireframe einer GUI, mit der sich eine Namensliste erstellen lässt. (20)



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

- Klickt ein Nutzer auf **Hinzufügen**, wird der Text aus dem Eingabefeld (lineEdit) zur Liste hinzugefügt.
- Klickt ein Nutzer auf **Löschen**, wird der ausgewählte Eintrag aus der Liste gelöscht.
- Der Löschen-Button ist genau dann klickbar, wenn ein Eintrag in der Liste ausgewählt ist.
- Beim Starten der Anwendung befinden sich die Einträge “Alice”, “Bob” und “Charles” in der Liste.

Tipp: Die folgenden Funktionen könnten für die Lösung wichtig werden:

- Das Signal `QListWidget::itemSelectionChanged()` wird ausgelöst, wenn sich eine Auswahl in der Liste ändert.
- `QListWidget::currentRow()` gibt den Index der aktuell ausgewählten Zeile aus, oder -1, wenn keine Zeile ausgewählt ist.
- `QListWidget::addItem(QString)` fügt einen String der Liste hinzu.
- `QListWidget::takeItem(int)` löscht den Eintrag mit dem übergebenen Index aus der Liste.
- Darüber hinaus dürfen Sie die Referenz im Anhang verwenden.

Trennen Sie den Quelltext in Header- und Source-Datei sinnvoll auf. Vervollständigen Sie den Code auf den folgenden Seiten.

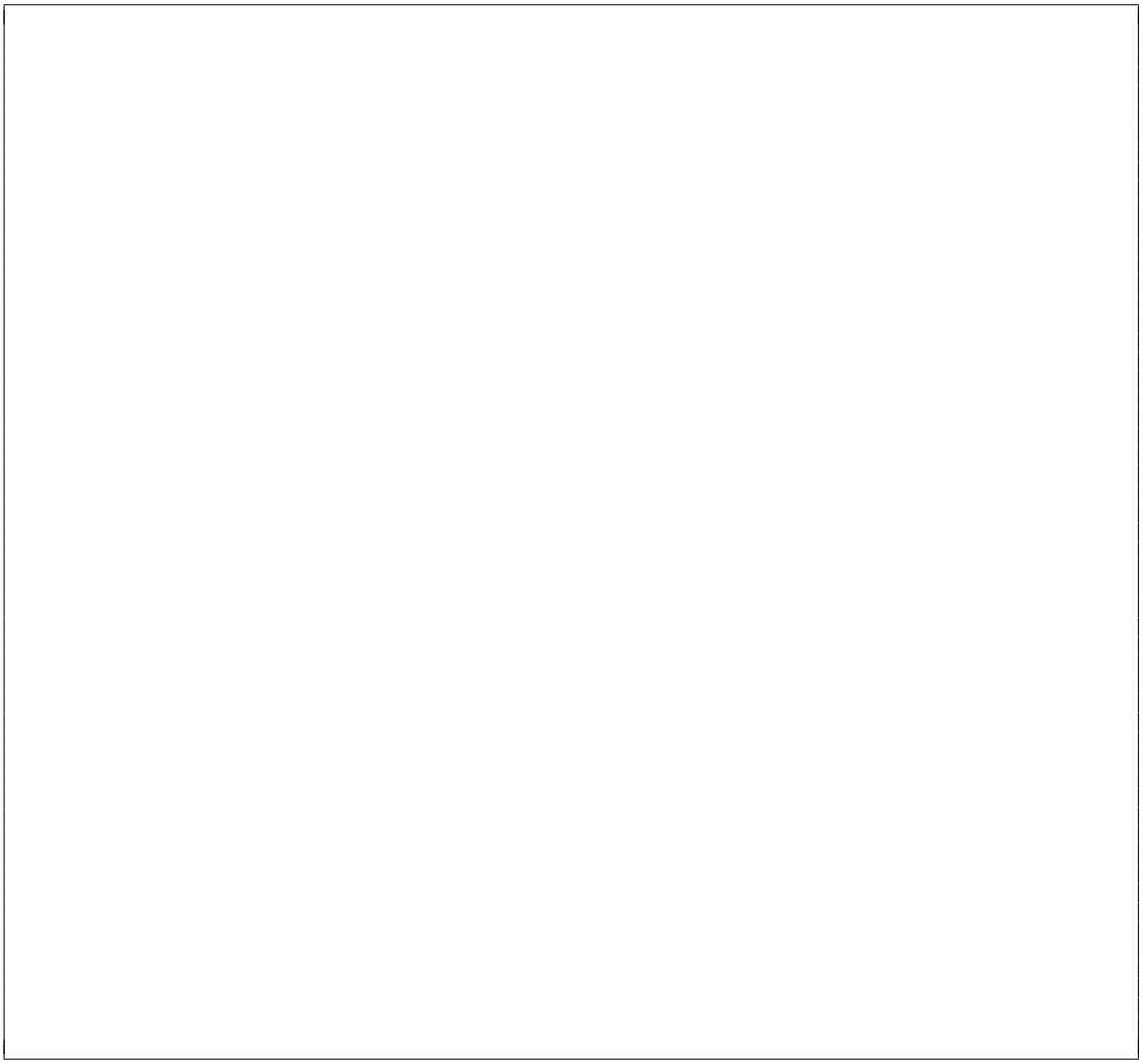
Headerdatei mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QLineEdit>
#include <QPushButton>

class MainWindow : public QMainWindow {
   Q_OBJECT

public:
   MainWindow(QWidget *parent = nullptr);
```



```
};

#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)

10. Erläutern Sie die Funktionsweise eines *Shared Pointer*. (5)

Analyse

11. Erläutern Sie die Fehler der folgenden beiden Code-Auszüge:

(a) (5)

```
1 int main() {
2     unique_ptr<int> ptr1(new int);
3     unique_ptr<int> ptr2 = std::move(ptr1);
4 
5     *ptr1 = 5;
6     *ptr2 = 8;
7 
8     return 0;
9 }
```

(b) (5)

```
1  QMutex mutex;
2
3  void threadSafeFunction() {
4
5      QMutexLocker* locker = new QMutexLocker(&mutex);
6
7      doComplexComputation();
8
9  }
```

Qt Cheat Sheet

Signale

QPushButton / QAbstractButton

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

QLineEdit

```
void    textChanged(const QString &text)
void    editingFinished()
void    inputRejected()
void    returnPressed()
void    selectionChanged()
```

Methoden

QLineEdit

Text auslesen
QString text() const

Text setzen
void setText(const QString &)

QAbstractButton

Auslesen: Ist Checkbox angehakt?
bool isChecked()

QString

Länge ermitteln
int size() const

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()
```

QLineEdit

```
void    clear()
void    copy() const
void    cut()
void    paste()
void    redo()
void    selectAll()
void    setText(const QString &)
void    undo()
```

Layout-Manager

QHBoxLayout
QVBoxLayout
QFormLayout
QBoxLayout
QGridLayout
QStackedLayout

Thread-Synchronisation

QMutex
QReadWriteLock
QSemaphore

Message-Boxen

```
QMessageBox::critical(QWidget *parent, const QString &title, const QString &text,
    StandardButtons buttons = Ok, StandardButton defaultButton = NoButton);
```