Projektprüfung: Einführung in die Softwareentwicklung

Sommersemester 2020 · 03. August 2020 – 09. August 2020

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Regeln und Hinweise zur Bearbeitung

Bitte lesen Sie sich die untenstehenden Hinweise als erstes sehr genau durch, bevor Sie mit der Bearbeitung der Aufgaben beginnen!

Die Regeln und Hinweise wurden bereits vorab verfügbar gemacht. In der Zwischenzeit haben wir noch einige wenige, aber sehr wichtige Ergänzungen vorgenommen, die in **rot markiert** sind. Es gilt diese Fassung (es betriff den master-Branch für die Abgabe und die Verwendung von Code-Schnipseln zur API-Nutzung, z.B. von Stack-Exchange).

- Im Fach „Einführung in die Softwareentwicklung“ wird im Sommersemester 2020 eine Take-Home-Prüfung durchgeführt, in der ein kleines Softwareprojekt bearbeitet werden soll.

Selbstständige Bearbeitung

- **Die Bearbeitung der Aufgaben muss selbstständig erfolgen**. Sie dürfen keine dritten Personen um inhaltliche Hilfe bei der Lösung der Aufgaben bitten und keine Hilfe dritter annehmen. Sie dürfen gleichfalls anderen Teilnehmern der Prüfung keine Hilfestellung anbieten.
- **Ausnahme ist der offizielle Kanal „Fragen zur Abschlussprüfung“ in MS-Teams, der für alle Lehrenden und Studierenden der Vorlesung einsehbar ist**. Hier dürfen Fragen zum Verständnis der Aufgaben gestellt werden. Hilfe bei Lösungen wird nicht angeboten (keine Antwort). Fragen, die wesentliche Hilfen bei der Lösung preisgeben, gelten als Regelverstoß. Im Zweifelsfall fragen Sie Ihren Tutor zuvor per Email, ob die Frage zulässig ist (das sollte sehr selten vorkommen; es sollte eigentlich in fast jedem Fall offensichtlich sein, dass zulässige Fragen zum Verständnis der Aufgaben keine Lösungen vorwegnehmen).
- **Rein technische Hilfe ohne inhaltlichen Bezug zu den Aufgaben ist zulässig.** („Kann ich mal Dein Internet benutzen?“ ist kein Regelverstoß).
- **Es ist zwingend notwendig, zusammen mit Ihrer Lösung eine Selbstständigkeitserklärung abzugeben**. Drucken Sie hierzu die letzte Seite dieses Dokumentes aus (oder schreiben Sie es handschriftlich ab, falls Sie keinen Drucker besitzen), unterschreiben Sie es (mit Ort und Datum) und scannen Sie das Ergebnis ein (ein Foto mit dem Smartphone ist auch ok, solange alles gut lesbar ist). Sie müssen in diesem Formular die u.a. die selbständige Bearbeitung und Ihre Prüfungsfähigkeit (keine gesundheitlichen Beeinträchtigungen) bestätigen. Heben Sie das Original unbedingt für Rückfragen auf. Die Erklärung muss in der Abgabe (via Gitlab) als Datei „`Selbstständigkeitserklärung.jpg`“ im Stammverzeichnis enthalten sein (andere Dateiformate wie PNG, GIF und PDF o.ä sind ebenfalls zulässig; achten Sie aber darauf, dass die Datei nicht zu groß wird – dies kommt sehr schnell bei Dateiformaten wie BMP, TIFF und RAW vor). Klausuren ohne Selbstständigkeitserklärung werden nicht korrigiert. **Liegt die Erklärung zur Korrektur nicht vor, so wird die Abgabe als „nicht bestanden“ bewertet.**

Zulässige Hilfsmittel

- **Der abgegebene Programcode muss in C++ vorliegen, und zwar so, dass er mit einem aktuellen Compiler, der den Sprachstandard C++-17 beherrscht** (GCC, Clang, MS-Visual C++, o.ä. in einer aktuellen Version) **übersetzt werden kann**. Mit den Standardpaketen aus Ubuntu 20, Linux Mint 20 (Paketname „build-essential") oder den aktuellen Downloads für Windows (Visual Studio / MS Build Tools for C++) gehen Sie auf Nummer sicher. Sollten Sie einen anderen als einen der genannten Compiler verwenden, fragen Sie kurz im Forum an.
- **Es ist zwingend Notwendig das verwendete Buildskript mit abzugeben.** Zulässig sind `CMake`, `Make` und `QMake`, in begründeten Ausnahmefällen auch Batchdateien (siehe unten). Sollten wir nicht in der Lage sein, Ihre Abgabe zu übersetzen, so kann dies zu erheblichen Abzügen führen.
- **Die Standardbibliotheken von C++** (also alles aus dem Namespace `std` und alle Teile des aktuellen C++-Sprachstandards) dürfen grundsätzlich frei verwendet werden.
- **Die Bibliothek Qt** (alle Versionen 5.x.x) **dürfen ebenfalls verwendet werden**, allerdings wird dabei die Verwendung **spezieller Qt-Komponenten ausgeschlossen**, da ansonsten die Schwierigkeit der Aufgabenstellung stark verzerrt würde. Details hierzu finden Sie in den konkreten Aufgabenstellungen.
- Wir empfehlen, ein Standardpacket von Qt zu verwenden, um Kompatiblitätsprobleme zu minimieren (z.B.: Wenn Sie Ubuntu 20 nutzen, sind die Standardpackete `qt5-default` / `qtbase5-dev-tools` auf dem Stand vom Version 5.12).
- Die wörtliche Übernahme von **Lösungen von Übungsaufgaben** (auch der **eigenen Gruppe**) ist **nicht erlaubt**, da es sich formal um eine Gruppenarbeit handelt. Sollten Sie sich bei der Reimplementation stark daran orientieren, dann ist das verständlich und zulässig, erfordert aber eine Quellenangabe (s.u.).
- Sie dürfen beliebig im Internet recherchieren und Bücher und Vorlesungsunterlagen aller Art nutzen. **Sie müssen allerdings alle *wesentlichen* Quellen, die sie verwendet haben, angeben:**
  - Einfaches Faktenwissen (z.B. Qt-APIs, C++-Sprachkonstrukte und Standardbibliotheken) kann ohne weiteres genutzt werden. Gleiches gilt für Standardalgorithmen aus dem Grundstudium (Sie brauchen Quicksort oder Dijkstra's Graphalgorithmen nicht zitieren, auch wenn Sie es in einem Lehrbuch nochmal nachgeschlagen haben).
  - Sollten Sie komplexere Ideen übernehmen (z.B. Architekturvorschläge oder Ihr Code orientiert sich an Pseudo-Code aus dem Internet), dann muss die Quelle angegeben und der übernommene Inhalt kurz beschrieben werden.
  - Wenn sich Code an früheren Lösungen von Übungsaufgaben orientiert, geben Sie dies bitte auch an. Eine wörtliche Kopie ist, wie gesagt, trotzdem nicht erlaubt (da

es sich formal um Gruppenarbeit handelt, auch wenn Sie alleiniger Autor sein sollten). Implementieren Sie die Lösung hier immer nochmal neu. Allerdings wird hierbei akzeptiert, wenn der neue Code starke Ähnlichkeiten zur Übungsversion aufweist, wenn eine Quellenangabe vorliegt (ohne Abzüge).

  - **Schreiben Sie dazu entsprechende Hinweise in eine Datei „Quellen.txt" im Stammverzeichnis Ihres Projektes.** Fehlt diese Datei, so wird angenommen, dass keine externen Quelle genutzt wurden, die über das Nachschlagen von Faktenwissen hinausgehen. Zusätzlich muss jeder Code, der auf externen Quellen basiert, mit Kommentaren im Code klar gekennzeichnet und abgegrenzt werden (Beispiel `/* Start Message-Loop nach Meier et al. */` …. `/* Ende Message Loop */`)

- **Das wörtliche Kopieren von fremden Code ist grundsätzlich verboten.** Auch Übersetzungen und Abwandlungen gelten als Plagiat, wenn die Quellenangabe dies nicht klar und deutlich kennzeichnet, und genau erklärt, was der eigene Beitrag ist. Positiv bewertet wird nur der eigene Beitrag – wir empfehlen daher, auf alle Übernahme von Varianten / Abwandlungen zu verzichten.

  - Code, der von einer Standardentwicklungsumgebung (Qt-Creator, CLion, Visual Studio) oder Build-Tools (z.B. moc) automatisch erzeugt wird, ist selbstverständlich grundsätzlich erlaubt und muss nicht besonders gekennzeichnet werden. Sollten Sie ein Tool benutzen, welches stark darüber hinaus geht (z.B. ein „Round-Trip UML Tool", welches C++-Code aus Diagrammen erzeugt, oder einen Interfacebuilder jenseits der Qt-Standardtools, so geben Sie dies bitte als Hilfsmittel an).

  - Der Qt-Interface-Designer (`qtdesigner` bzw. die UI-Designer-Komponente im `qtcreator`) dürfen ohne Einschränkungen benutzt werden. Es ist nicht verlangt, alle GUIs per Hand zu programmieren.

  - Codeschnipsel, die lediglich die Verwendung von Bibliotheks APIs (Schnittstellen von Qt und den C++-std-Bibliothekn) erklären (z.B. Stackoverflow Antwort zu „Wie erzeuge ich Qt Menü-Optionen, bei denen immer nur eine ausgewählt werden kann?" - https://stackoverflow.com/questions/23429663/qt-mutually-exclusive-checkable-menu-items). Dürfen kopiert und angepasst werden, wenn eine Quellenangabe gemacht wird. Dies gilt nur für Fragen der API-Nutzung; komplexere Algorithmen sind nicht erlaubt! Das Urherberrecht gilt ebenfalls weiterhin.

  - Sie dürfen für Ihr GUI Icons benutzen, die aus freien Quellen stammen (die also die Nutzung in diesem Rahmen rechtlich erlauben), wenn Sie dies in den Quellenangaben kenntlich machen.

Abgabe

- **Abgaben erfolgen via Gitlab** (`gitlab.rlp.net`)
  Sie werden dazu in eine neue Gruppe namens „`eis20-projekt`“ eingeladen. In dieser Gruppe befindet sich das Abgaberepository, in das nur Sie selbst, David Hartmann und Michael Wand Zugriff haben. Sollte sich kein Repository in dieser Gruppe befinden, teilen Sie dies bitte umgehend David Hartmann per Microsoft Teams mit.
- **Die Abgabe muss als commit in den Masterbranch erfolgen. Andere Branches als „master“ können aus technischen Gründen nicht bewertet werden.**
- **Anonyme Abgabe:** Die Aufgaben werden anonymisiert korrigiert. Achten Sie daher darauf, dass in Ihrem Quellcode keine Angaben zu finden sind, die Sie als Person offensichtlich identifizieren (keine Namen).
- **Notfallplan:** Sollte durch einen unerwarteten Notfall der Gitlab-Server des Landes RLP nicht verfügbar sein oder nicht reagieren, so ist auch das folgende Vorgehen zulässig:
  - Packen Sie Ihre Abgabe (komplett, mit allen Dokumenten, Code und Daten, die sonst im Gitlab erscheinen sollten) in ein ZIP-Archiv.
    Wichtig: Nur die in dem Archiv enthaltenen Dateien und Informationen zählen – alles andere wird ignoriert.
  - Bestimmen Sie eine MD5-Prüfsumme von dem Archiv. Dazu gibt es verschiedene Tools (in der Regel für die Kommandozeile):
    - Linux: `md5sum` (Standardtool in den meisten Distributionen)
      Kommandozeile: `md5sum <name-der-abgabe.zip>`
    - Windows: `CertUtil` ist seit Windows 7 im Lieferumfang:
      Kommandozeile: `certutil -hashfile <name-der-abgabe.zip> MD5`
    - Mac: `md5` scheint ein Standardtool zu sein
      (ohne Gewähr; wir besitzen keine Macintosh-Geräte)
  - Senden Sie nun diese Prüfsumme an David Hartmann via Email (dahartma@uni-mainz.de). Der Zeitpunkt, an dem diese Email ankommt, gilt als Zeitpunkt der Abgabe.
  - Behalten Sie unbedingt das original ZIP-Archiv. Lassen Sie es Herrn Hartmann mit einem Dateiaustauschdienst, z.B. seafile.rlp.net am nächsten Tag zukommen. Schicken Sie niemals ihre vollständigen Lösungen per Email!
  - **Wichtig!** Unbedingt nur MD5-Hashes per Email versenden. Schicken Sie niemals das ganze Archiv via Email! Dies kann das Email-System belasten, so dass Ihre Abgabe oder die Ihrer Kommilitonen nicht (rechtzeitig) ankommt.
  - Abgaben, in denen der Code selbst per Email eingereicht wird, gelten als nicht bestanden!

- Wegen des hohen Aufwandes für die Betreuer dürfen Sie den Notfallplan auch **nur im *Notfall* benutzten**, d.h., falls es Ihnen aus technischen Gründen unmöglich ist, die Abgabe via Gitlab fristgerecht zu tätigen. Neben einem Ausfall auf JGU/RLP-Seite zählt dazu auch die Möglichkeit, dass z.B. Ihr eigenes Internet ausfällt und Sie nur noch via Smartphone eine kurze Nachricht schicken können. In jedem Fall ist eine Begründung (via Email) erforderlich.

Zeitplan

- Die Aufgaben werden am **Montag, den 03. August 2020 um 10:00 MESZ** auf den **Webseiten zur Vorlesung** sowie dem **JGU-Reader** veröffentlicht**.**
- Die Lösungen müssen bis **spätestens Sonntag, den 09. August 2020 um 23:59h MESZ** auf gitlab.rlp.net in Ihrem Projektverzeichnis zur Klausur **im Masterbranch** eingecheckt werden.
  - Für Studierende mit überlappenden Klausuren kann eine Verlängerung der Bearbeitungsfrist beantragt werden. Dies muss **vorab** (vor dem 03.08.2020) per Email oder Teams an David Hartmann (dahartma@uni-mainz.de) erfolgen. Sollten Sie eine Rückmeldung erhalten haben, dass eine Verlängerung gewährt wurde, so gilt diese Abgabefrist.
  - Spätere Abgaben werden nur mit reduzierter Punktzahl gestattet. Dabei reduziert sich die Punktzahl stundenweise linear von 0% bis 50% der Punkte über einen Zeitraum von 12h nach Abgabefrist. **Danach ist ein Bestehen nicht mehr möglich**, und die Abgabe gilt auch ohne Korrektur als nicht bestanden. Eine spätere Abgabe wird nur berücksichtigt, wenn dies explizit gewünscht wird (Email an David Hartmann). Zulässig sind nur Abgaben im oben definierten Sinne (Gitlab commits oder email). Es gilt der Zeitstempel[1] des Empfängers (Email bzw. Gitlab).
  - **Tipp:** Sie können gerne mehrere commits im Gitlab vor der Abgabefrist machen; es zählt dann automatisch die aktuellste commit im Masterbranch vor der Frist. Auf dem Wege können Sie Risiken (Datenverlust, Ausfall von Hard- oder Software) verringern. Machen Sie möglichst nur Abgaben nach Ablauf der Frist, wenn Sie die Option mit den verringerten Punkten tatsächlich nutzen möchten (es macht den Korrektoren sonst vermeidbare zusätzliche Arbeit).

[1] Zeitstempel werden automatisch über künstliche, stündlich angelegte Branches erzeugt, die auf dem Masterbranch basieren (daher können keinen anderen Branches bewertet werden). Wundern Sie sich also nicht, falls Sie nach der Abgabefrist ein git pull ausführen. Die Daten im Masterbranch werden dadurch aber in keinem Fall verändert.

Weitere Regelungen

- Alle weiteren, im Vorfeld via Reader-Email vom Dozenten versandten Regularien gelten weiterhin. Die Angaben auf diesem Aufgabenblatt dienen zur Erinnerung und Vervollständigung. Neu ist insbesondere die Einschränkung auf den Masterbranch bei der Abgabe – beachten Sie dies unbedingt; andere Branches werden grundsätzlich nicht gewertet.
- Behalten Sie den MS-Teams Kanal „Ankündigungen zur Abschlussprüfung“ selbstständig im Auge. Jede Nachricht, die alle erreichen soll, wird zwar gleichzeitig über den Reader verschickt, erfahrungsgemäß bringt dieser jedoch zum Teil sehr hohe Latenzen mit sich.

Wir wünschen Ihnen nun viel Erfolg bei der Bearbeitung der Aufgaben!

Übersicht Aufgabenstellung

Als erstes möchten wir eine Übersicht über die Aufgabenstellung geben. Danach werden die insgesamt drei Ausbaustufen des Projektes im Detail spezifiziert. Am Ende folgen die Bewertungsmaßstäbe, die auch den Korrektor*innen bei der Begutachtung Ihrer Lösungen vorliegen.

Zur besseren Anschauung steht auch eine Binärversion der Musterlösung in allen drei Ausbaustufen zur Verfügung[2].

Aufgabe: **Programmieren Sie ein Vektorzeichenprogram**
(Ähnlich zum Beispiel in der Vorlesung aus LE09: „S13-Muster-GUIs")
*Insgesamt können **100** Punkte erzielt werden. 50 Punkte reichen auf jeden Fall aus, um zu bestehen.*

Grundausbau: Für die Basisversion müssen Sie eine graphische Anwendung mit Qt programmieren, in der Sie mit der Maus ein oder mehrere Polygone (geschlossene, zusammenhängende Linienzüge) erstellen können.
*Eine vollständig korrekte Lösung gibt **40** Punkte.*

Ausbaustufe 1: Fügen Sie Funktionalität hinzu, um die gemalten Polygonzüge zu speichern und wieder zuladen (als Linienzüge, nicht als Bilder).
*Eine korrekte Lösung gibt **10** Punkte. Für alle weiteren Werkzeuge und Formen in Ausbaustufe 2 wird überprüft, dass die Lade- und Speicherfunktionen auch funktionieren; sonst gibt es dort weniger Punkte.*

Ausbaustufe 2: Implementieren Sie jeweils Werkzeuge, um Texte und andere Formen (zur Auswahl stehen Ellipsen/Kreise, Rechtecke/Quadrate und Sterne) zu der Zeichnung hinzuzufügen.
*Eine (vollständig korrekte) Lösung, die neben dem Textwerkzeug noch ein weiteres Werkzeug implementiert, sowie auch die Funktionen zum Laden und Speichern, wird mit der Maximalpunktzahl von **20** Punkten bewertet. Sollte mehr als ein weiteres (nicht-Text)-Werkzeug implementiert werden, so gilt die beste Lösung.*

Ausbaustufe 3: Erlauben Sie dem Benutzer, den Bildausschnitt und die Vergrößerung frei zu wählen (Zoom / Pan).
Tipp: Die Ausschnittwahl sollte man schon beim Entwurf im Auge haben.
*Eine vollständig korrekte Lösung gibt **15** Punkte..*

Gestaltung: Ob die Anwendung insgesamt schön gestaltet ist und einfach zu benutzten, wird mit 5 zusätzlichen Punkten bewertet (genaue Kriterien sind definiert). Dazu zählt ein einfaches Hilfesystem, Toolbars und Menükommandos für Werkzeuge, und konsistente Bedienung (Maus-/Tastaturbefehle).
*Eine gut gestaltete Lösung mit allen Ausbaustufen erhält bis zu **5** Punkte. Besonders gelungene zusätzliche Features, jenseits der Aufgabenstellung,*

[2] Es muss sicher nicht gesagt werden, dass die Anwendung von Reverse-Engineering Werkzeugen auf die Binärdateien der Musterlösung genauso als Täuschungsversuch zählt. Es fällt auch ziemlich sicher auf, denn wir kennen unseren eigenen Code sehr gut.

können im Ermessen der Korrektoren, zusätzlich mit bis zu 5 Bonuspunkten belohnt werden.

Abgabe: Die Abgabe muss sich direkt auf der Kommandozeile aus dem Quelltext übersetzen lassen. Zulässig sind Makefiles (für Make, QMake oder CMake), und in Ausnahmefällen Batchdateien für Ihr Betriebssystem (also `cmd/.bat` auf Windows, `bash` `bash-/zsh-Skripte` auf Linux und MacOS-X). Ihre Abgabe sollte eine Datei README.txt enthalten, in der beschrieben steht:

- Welches Betriebssystem Sie genutzt haben
- Welche Version von Qt (5.x.x) und welchen C++-Compiler (mit Versionsnummer[3]) sie benutzt haben.
- Eine Anleitung, wie man Ihren Quellcode übersetzt

*Eine korrekte Lösung erhält **5** Punkte. Sollte es uns nicht gelingen, das Programm (überhaupt) zu übersetzen (auch mit Nachbesserungen durch die Korrektoren/innen), so führt dies allerdings zu erheblicher Abwertung. Eine gute Lösung ist daher hier wichtig!*

Diskussion: Stellen Sie eine Datei „`Architektur.txt`“ im Hauptverzeichnis ihres Projektes bereit, und diskutieren Sie darin, wie Sie die Architektur Ihrer Anwendung gestaltet haben, und warum dies aus Ihrer Sicht so sinnvoll ist.
*Eine gute Architektur mit überzeugender Erklärung erhält bis zu **5** Punkte.*

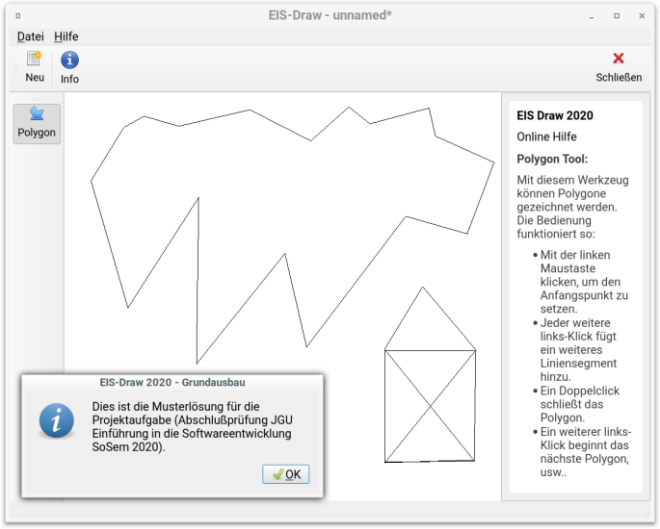
Wichtig – Einschränkung bezüglich Qt: Die Bibliothek Qt bietet bereits weitgehende Unterstützung für Vektorgraphik, die hier natürlich nicht benutzt werden darf; dies gilt für alle Ausbaustufen. Konkret ist die Benutzung der Klassen aus dem Qt Paket „`Graphics View Framework`“ (also u.a. `QGraphicsScene`, `QGraphicsItem` mit allen Nachfahren) nicht gestattet. Gleiches gilt für die – ähnlich angelegte – Teilbibliothek „`Qt Quick Scene Graph`“, also z.B. `QSGNode` oder `QQuickItem` mit allen Nachfahren, sowie für das gesamte `Qt3D` Paket. Sollten diese Komponenten fälschlicherweise verwendet werden, **werden pauschal 25 Punkte von der Lösung abgezogen**. Sie müssen die Klassen, die die Zeichenwerkzeuge und die Zeichnungselemente repräsentieren, und im richtigen Bildausschnitt auf die Zeichenfläche bringen, vollständig selbst implementieren.

Benutzt werden dürfen und sollen `QPainter` (zum Zeichnen; alle Funktionalität dieser Klasse ist erlaubt) und die Standard Qt-GUI-Elemente, die nicht speziell für die Verwaltung von Vektorgraphik gedacht sind. Die Verwendung von mathematischen Funktionen, wie Vektor- und Matrix Klassen (`QPoint`, `QPointF`, `QMatrix`), und diversen Hilfsklassen (QString, QFile, etc.) ist natürlich auch erlaubt. Verboten sind nur die o.g. Scenegraphen-Frameworks (die normale GUI-Elemente der „`QWidget`“s fallen natürlich auch nicht darunter).

Im Folgenden erklären wir die Aufgabenstellungen und deren Bewertung genauer.

[3] Beispiel: „gcc 9.3.0“, „Microsoft Visual C++ 2019“, „clang 10.0.0-4ubuntu1“ oder ähnliches.

Grundausbau: Ein interaktiver Editor für Zeichnungen aus Polygonen (40 Punkte)



Ihre Lösung muss die folgenden Eigenschaften haben:

- **Programfenster:** Beim Start des Programms muss ein Fenster mit einem geeigneten Hauptmenü erscheinen. Das Fenster muss frei in der Größe veränderbar sein. (5 Punkte)
- **Hauptmenü:** Das Menu sollte die Möglichkeit bieten, die Anwendung zu beenden („Datei/Schließen…“), die aktuelle Zeichnung zu löschen und eine neue, leere Zeichnung anzulegen („Datei/Neu…“) sowie eine Dialogbox zu öffnen, die den Namen des Programms nennt („Hilfe/Info…“). (10 Punkte)
- **Polygonwerkzeug:** Die Arbeitsfläche des Anwendungsfensters soll auf Mausklicks reagieren[4], und erlauben, geschlossene Linienzüge zu zeichnen. Beispielsweise können Sie den einfachen „Klick“ für das Hinzufügen eines weiteren Punktes verwenden, und den Doppelklick um das aktuelle Polygon abzuschließen und ein neues zu beginnen. Die gezeichneten Polygone (oder andere Elemente, aus späteren Ausbaustufen) müssen in dem Fenster angezeigt werden, und auch dann noch sichtbar sein, wenn das Fenster zwischenzeitlich verdeckt war, oder in der Größe verändert wurde. Wird gerade ein Liniensegment gezeichnet, so soll es dem Benutzer als Vorschau angezeigt werden (dem Mauscursor folgend). (25 Punkte)

[4] Wenn Sie weitere Ausbaustufen implementiert haben, kann es sein, dass erst ein Werkzeug ausgewählt werden muss, bevor diese Funktion zur Verfügung steht. Das Programm muss also nicht im „Polygonmodus“ starten.

Hinweise zur Bewertung (unvollständig):

Programfenster

- -2 Punkte falls nicht in der Größe veränderlich. Reaktion des Programms auf Größenänderung freigestellt, solange die Zeichnung bzw. ein sinnvoller Ausschnitt weiterhin sichtbar sind.

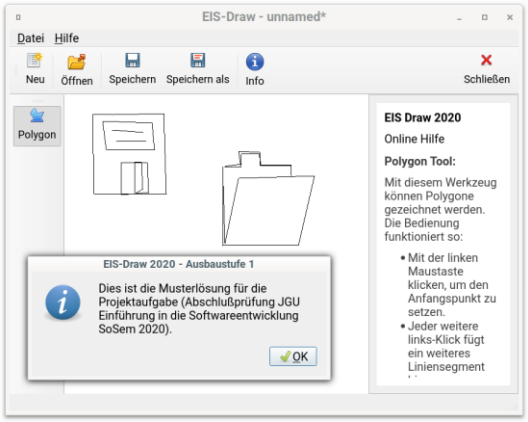
Hauptmenu

- 2 Punkte für Menu als solches
- 1 Punkte für Schließen, 2 Punkte für „Info“, 5 Punkte für „Neu“
- Die Aufgabe ist so gedacht, dass man ein Klassisches Pull-Down-Menu implementieren soll, wie es die meisten Anwendungen seit 2-3 Jahrzehnten nutzen. Wird stattdessen eine äquivalente Funktionalität auf anderem Wege realisiert, z.B. mit einem „Ribbon“-Design wie bei Microsoft Office, oder einer gut durchdachten Palette an Toolbars, so kann ebenfalls die volle Punktzahl erreicht werden. Die Musterlösung zeigt exemplarisch eine Kombination eines klassischen Pull-Down Menüs mit Toolbar.

Zeichenwerkzeuge

- Fehler in der Ereignisbehandlung beim Polygonzeichnen: zwischen -2 (hakt in Spezialfällen) bis -20 Punkte (Zeichnen mehrerer Polygone gar nicht möglich). Kann nur ein Polygon gezeichnet werden, so werden maximal 5 Punkte vergeben (kann nicht geschlossen werden) bzw. 8 Punkte (kann geschlossen werden, aber das nächste startet nicht).
- Die Vorschaufunktion gibt 5 Punkte, aber nur einen Punkt, falls die Linien nachziehen/beim Bewegen der Maus nicht gelöscht werden.

Ausbaustufe 1: Laden und Speichern (10 Punkte)



Ihre Lösung muss die folgenden Eigenschaften haben:

- **Laden/Speichern:** Das Programm sollte die Möglichkeit bieten, die Zeichnungen zu speichern und wieder zu laden. Das Dateiformat dürfen Sie dabei frei festlegen (gerne ein selbst erfundenes). Die Funktionen sollten als Menüpunkte (und/oder via Toolbar oder einem entsprechenden GUI-Element) zur Verfügung stehen. (6 Punkte)
- **Komfort:** Ihr Programm sollte Benutzerfehler wie folgt abfangen: Wenn es beendet wird, obwohl man die Zeichnung inzwischen geändert hat, so soll eine Sicherheitsabfrage den Benutzer davor warnen, und die Möglichkeit geben, das Beenden zu überdenken. Das gleiche gilt, falls eine neue Zeichnung geöffnet wird, die eine veränderte, zuvor angelegte Zeichnung überschreibt (Falls Ihre Anwendung immer ein neues Fenster für neue Dokumente öffnet, so gilt dies auch als erfüllt). Eine Warnung beim Überschreiben auf dem Datenträger bereits vorhandener Dateien sollte ebenfalls erfolgen. Sofern der Benutzer die Zeichnung gar nicht geändert hat, sollte keine Speicherwarnung erschienen: Das heißt, die Anwendung muss verfolgen, ob inzwischen Änderungen vorgenommen wurden. (4 Punkte)
- **Hinweis:** Das Laden und Speichern wie auch die Komfortfunktionen müssen auch für die Features der weiteren Ausbaustufen implementiert werden, um die volle Punktzahl in den späteren Ausbaustufen zu erreichen! Man muss also in der Lage sein, die Formen, die mit alle anderen Werkzeugen erstellt wurden, richtig zu speichern und laden, und muss Veränderungen an der bearbeiteten Zeichnung richtig nachverfolgen.

Hinweise zur Bewertung (unvollständig):

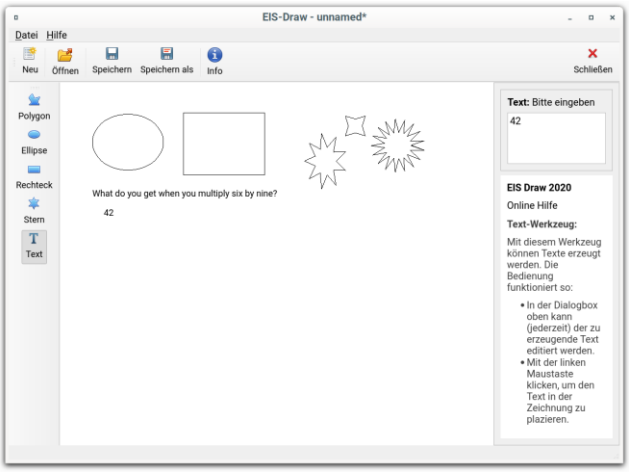
Laden und Speichern

- Für das Speichern und Laden gibt es jeweils 3 Punkte.
- Sollte das Wiedereinladen nicht funktionieren, so gibt es insgesamt nur maximal 2 Punkte, sollte es nicht zuverlässig funktionieren, so gibt es maximal 3 Punkte.
- Es muss möglich sein, eine leere Szene zu speichern und zu laden (-2 Punkte sonst).

Komfort

- Je $\frac{1}{2}$ Punkt für Sicherheitsabfrage bei Schließen/Beenden, Öffnen, Überschreiben, Neue Zeichnung.
- 2 Punkte für korrekte Handhabung der Modifikation (keine Abfrage falls nichts geändert wurde).

Ausbaustufe 2: Zusätzliche Zeichenwerkzeuge (20 Punkte)



Fügen Sie Ihrer Lösung die folgenden Features hinzu:

- **Ein zusätzliches Zeichenwerkzeug:** Zur Auswahl stehen
  - Ellipsen/Kreise
  - Rechtecke/Quadrate oder
  - Sterne

  Die Objekte müssen sich mit der Maus in die Zeichnung einfügen lassen (Klick auf Mittelpunkt, „aufziehen" der Größe; siehe Beispiel in der Musterlösung). Dabei soll eine Vorschau angezeigt werden (Form folgt der sich bewegenden Maus).
  Eine Abgabe, bei der die Dimensionen/Positionen etwa per Textfeld übergeben werden ist nicht zulässig.

  Bei Ellipsen und Rechtecken muss eine Symmetrie-Option bestehen, mit der erzwungen wird, dass ein Kreis oder ein Quadrat gezeichnet wird (z.B. muss man in der Musterlösung die Steuerungstaste während des Aufziehens gedrückt halten, um ein Quadrat bzw. einen Kreis zu erzwingen). Bei Sternen muss die Anzahl der Zacken des Sterns einstellbar sein (auf irgendeine Art und Weise, aber sinnlose Eingaben müssen unterbunden werden); eine Symmetrie-Option ist nicht nötig (die Musterlösung implementiert diese trotzdem).

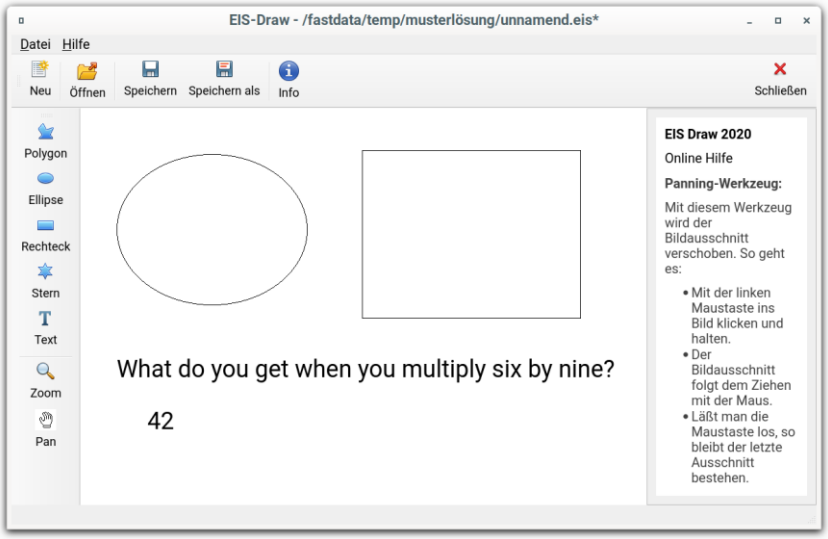
  Werden mehrere Werkzeuge implementiert, so zählt die beste Lösung (die Musterlösung implementiert alle drei Werkzeuge; dies ist aber nicht erforderlich). (10 Punkte)

  Ein Textwerkzeug: Hierzu wird zunächst ein Text eingegeben (z.B. in eine Dialogbox) und danach in der Zeichnung platziert. Wird die Szene später vergrößert oder verkleinert (siehe Ausbaustufe 3), so soll der Text proportional mitvergrößert werden (und am selben Ort verbleiben). (10 Punkte)

Hinweise zur Bewertung (unvollständig):

- Das Laden und Speichern der neuen Zeichenelemente aus Ausbaustufe 1 ist Teil der Aufgabenstellung. Hierfür werden jeweils 2 (Laden) + 2 (Speichern) Punkte vergeben (also insgesamt 8 von 20 Punkten).
- Für die korrekte Detektion von Modifikationen durch die neuen Zeichenwerkzeuge wird 1 Punkt vergeben.
- Wie in der Grundausbaustufe sollen auch die neuen Werkzeuge eine Vorschau der gezeichneten Primitive anzeigen („Aufziehen" der Ellipse/Rechtecke/Sterne bzw. Platzieren des Textes). Hierfür gibt es jeweils einen Punkt pro Werkzeug.
- Die Symmetrieoption von Kreis und Ellipse bzw. das Einstellen der Anzahl von Zacken beim Sternwerkzeug wird mit 2 Punkten bewertet.
- Wenn das Rendering der neuen Formen nicht funktioniert (die erstellten Zeichnungselemente erscheinen nach dem Erstellen nicht in der Szene, oder verschwinden, wenn das Fenster verdeckt und neugezeichnet oder in der Größe verändert wird, so können maximal 3 Punkte pro Werkzeug erreicht werden.

Ausbaustufe 3: Zoom & Pan – echte Vektorgraphik (15 Punkte)



Der Witz bei der „Vektorgraphik“ ist natürlich, dass man beliebig in eine Zeichnung hereinzoomen kann, ohne dass man Pixel (oder andere Artefakte sieht). Implementieren Sie daher:

- **Ein Zoomwerkzeug:** Der Benutzer soll in der Lage sein, den Vergrößerungsmaßstab der Zeichnung (Umrechnung von Weltkoordinaten in Pixelkoordinaten) mit der Maus stufenlos (d.h. bis auf Mausauflösung) einzustellen. Dabei soll auch der Text entsprechend mit vergrößert und verkleinert werden (eine Rundung auf eine Genauigkeit von 1 Pixel/Buchstabe ist ausreichend; Qt bietet von Haus aus nur ganzzahlige („`int`“) Fontgrößen; man sieht in der Musterlösung, dass die Schriftgröße beim Zoomen etwas springt). (8 Punkte)
- **Werkzeug zur Verschiebung des angezeigten Bildausschnittes:** Der Benutzer soll in der Lage sein, den angezeigten Ausschnitt der Zeichnung mit der Maus (oder mit Scrollbalken) stufenlos zu verschieben. (7 Punkte)

Alle Werkzeuge müssen auch im neu eingestellten Ausschnitt konsistent funktionieren (also neue Objekte an den richtigen Koordinaten anlegen).

Die Steuerung von Zoom und Verschiebung muss stufenlos (bis auf die Genauigkeit der Mausbewegung in Pixeln) mit der Maus möglich sein. Hierzu können z.B. alternativ implementiert werden:

- „click & grab“-Werkzeuge, bei denen man das Bild „anfasst“ und verschiebt oder zoomt.
- „Ausschnitts“-Werkzeuge, bei denen man den Bildausschnitt mit der Maus aufzieht (zum Reinzoomen + Ausschnitt festlegen) und mit einer anderen Funktion wieder herauszoomt.
- Scrollbalken, an denen man zum Verschieben ziehen kann, mit einer geeigneten Zoomfunktion (z.B. auch ein Schieberegler).

Die Gestaltung ist auch jenseits dieser Auswahl freigestellt; für volle Punktzahl wird aber erwartet, dass der Benutzer die Funktionen sinnvoll und zielgerichtet nutzen kann (schwer beherrschbare Reaktionen führen zu einer Abwertung).

Hinweise zur Bewertung (unvollständig):

- Verschiebung wird mit 7, Zoom mit 8 Punkten bewertet.
- Für die volle Punktzahl muss es möglich sein, Verschiebung und Skalierung mit der Maus im o.g. Sinne stufenlos einzustellen. Dabei muss die Anwendung sich konsistent verhalten (eine „click & grab"-Funktion, bei der man mit der Maus verschiebt, muss das Bild so bewegen, wie die Maus bewegt wird, nicht langsamer oder schneller). Für Inkonsistenz können jeweils 3 Punkte abgezogen werden.
- Bei schwer beherrschbarer Interaktion können noch mehr Punkte abgezogen werden. Wenn der Bildausschnitt z.B. willkürlich umherspringt, können 0 Punkte vergeben werden.
- Wenn die anderen Werkzeuge mit neuem Ausschnitt nicht mehr funktionieren, so wird wie folgt bewertet:
  - Korrekte Darstellung der Szene 3 bzw. 4 Punkte (Zoom/Verschiebung)
  - Korrekte Funktion der Zeichenwerkzeuge jeweils 2 (Zoom) und 2 (Verschiebung) Punkte für Grundausbau und Ausbaustufe 1.
  - Bei leichten Fehlern (leichter Offset / noch benutzbar) wird ein Punkt abgezogen.
  - Bei graduellen Fehlern (z.B. immer schlimmer mit mehr Zoom) werden 2 bzw. 3 Punkte abgezogen (Zoom/Verschiebung).
  - Katastrophale Fehler (nix geht mehr oder Absturz) mit 0 Punkten.
- Die Zoom+Pan Funktion muss mit allen Primitiven der vorangegangen Ausbaustufen funktionieren. Die Punkte werden dabei anteilig vergeben: Wird z.B. nur das Polygonwerkzeug implementiert, so können maximal 1/3 der Gesamtpunkte für diese Aufgabe erreicht werden. Bei zwei Werkzeugen 2/3 und bei nur bei drei Werkzeugen alle Punkte. Das Erreichen der vollen Punktzahl ist daher von der erfolgreichen Bearbeitung von Ausbaustufe 2 abhängig.

Gestaltung: Machen Sie das Programm leicht und angenehm zu benutzen! (5 Punkte + 5 Bonuspunkte)

Fügen Sie Ihrer Anwendung folgende Features hinzu (2+1+1+1 Punkte):

- Ein Hilfesystem, das für jedes Werkzeug anzeigt, wie das Programm zu bedienen ist. Zum Beispiel können Sie eine Infoleiste am unteren (oder rechten) Rand des Fensters anzeigen, das die aktuellen Mausbefehle kurz erklärt (z.B. bei Polygonwerkzeug: „Klicken für neuen Eckpunkt, doppelklicken für schließen/neues Polygon.“)
  - Sollte keine Hilfe zur Verfügung stehen, muss unbedingt in der `README.txt` Datei im Hauptverzeichnis die Bedienung erklärt sein. Erschließt sich die Bedienung der/dem Korrektor/in nicht, so wird die Aufgabe u.U. als nicht gelöst bewertet.
- Fügen Sie eine Toolbar (zusätzlich zum Hauptmenu) hinzu, über die die implementierten Werkzeuge ausgewählt werden können.
- Fügen Sie Keyboard-Shortcuts hinzu, mit denen sich die Funktionen des Programms ohne Maus anwählen lassen (z.B. Ctrl+S für Speichern, oder eine Taste für die Auswahl des Textwerkzeuges).
- Achten Sie darauf, dass die Bedienung konsistent ist (ähnliche Mausgesten für alle Werkzeuge, sinnvolle Tastatenkürzel, klar gegliederte Menus/Toolbars).

Features, die über die Anforderungen der Aufgabenstellung hinausgehen, könne mit bis zu 5 zusätzlichen Bonuspunkten bewertet werden.

Wichtig: Sorgen Sie immer dafür, dass Ihre Features bei der Benutzung des Programms direkt sichtbar und entdeckbar („discoverable“) sind; wenn die Korrektoren/innen sie beim Ausprobieren des Programms nicht finden können, gibt es auch keine Punkte.

Abgabe: Stellen Sie ein Build-Skript bereit (5 Punkte)

Ihre Abgabe muss sich direkt auf der Kommandozeile aus dem Quelltext heraus übersetzen lassen. Hierzu muss eine Builddatei (`Make`-File, `QMake/*.pro`-Datei, `CMake`-Datei(en)) bereitstehen, die sich per `qmake`, `cmake` und bzw. oder `make` übersetzen lassen.

Schreiben Sie in die Datei `README.txt` (s.u.) unbedingt genau rein, wie man Ihren Code übersetzt! Sollte sich dies nicht erschließen, und den Korrektoren/innen keine schnelle Lösung einfallen, so kann die Abgabe als nicht funktionsfähig gewertet werden (wodurch wesentlich mehr als 5 Punkte abgezogen werden!).

Sollte es Ihnen nicht gelingen, ein Make-Skript in einem der drei genannten Formate anzufertigen (welche übrigens von den üblichen IDEs automatisch generiert werden, was zur Lösung der Aufgabe ausreichend sein sollte), so können auch Batchdateien für Ihr Betriebssystem (also `cmd/.bat` auf Windows, `bash bash-/zsh-Skripte` auf Linux und MacOS-X) bereitgestellt werden. Gibt es einen Grund außerhalb Ihrer Verantwortung, so kann damit die volle Punktzahl erreicht werden, sonst maximal 3 von 5 Punkten. Erklären Sie daher ggf. kurz in der `README.txt` (s.u.) Datei, warum Sie diese Option gewählt haben.

Hinweis: Zur Korrektur werden wir virtuelle Maschinen mit frisch installiertem Linux (z.B. Ubuntu 20) oder Windows (8 oder 10) einsetzen, zusammen mit den Standardpaketen für gcc, clang (Linux) oder gcc, msvc (Windows) und dem aktuellen Qt in der 5er Version.

Drei Pflichtangabe: (schlimmstenfalls: nicht bestanden)

(1) Anleitung: Ihre Abgabe muss eine Datei `README.txt` im Hauptverzeichnis enthalten, in der beschrieben steht:

- Welches Betriebssystem Sie genutzt haben (Linux, Windows, Mac, mit Version; andere bitte vorher absprechen!)
- Welche Version von Qt wurde benutzt?
- Welchen Typ und welche Version des C++-Compilers haben sie benutzt?
- Wie man Ihren Quellcode übersetzt.
- Wie man Ihr Programm bedient.

Die ersten drei Punkte dienen dazu, dass die Korrektoren/innen bei Problemen selbstständig nachbessern können. Gelingt dies nicht aufgrund fehlender Angaben, liegt das Risiko auf Ihrer Seite. Die letzten beiden Angaben sind sehr wichtig, damit die Korrektur sinnvoll durchgeführt werden kann. Fehlen hier Angaben, so liegt das Risiko einer schlechteren Bewertung ebenfalls auf Ihrer Seite. **Fehlende Angabe können nicht in der Klausureinsicht nachträglich erklärt werden** (bewertet wird nur, was zum Korrekturzeitpunkt mit verhältnismäßigem Aufwand nachvollziehbar und erkennbar war. Nehmen Sie diesen Punkt bitte sehr ernst und wichtig, sonst klappt das alles nicht!).

(2) Quellen: Sollten sie Quellen von dritter Seite (wie Eingangs erklärt) benutzt haben, so müssen die in einer weiteren Datei „`Quellen.txt`“ aufgeführt sein. Optional ist auch erlaubt, die Quellenangaben in der Datei „`Readme.txt`“ zu integrieren, andere Angaben werden aber **grundsätzlich nicht**

akzeptiert. Das Fehlen von Quellenangaben in diesen beiden Dateien bringt zum Ausdruck, dass keinerlei zusätzliche Quellen benutzt wurden (mit allen rechtlichen Konsequenzen falls sich dies als unzutreffend herausstellen sollte). Ungeachtet dessen müssen Quelltextbestandteile aus Fremdquellen **zusätzlich** mit Kommentaren eindeutig gekennzeichnet werden (Start und Ende müssen klar erkennbar sein).

(3) Die Erklärung zur Prüfung muss in einer Datei „`Selbstständigkeitserklärung.jpg`" (oder in einem anderen Format wie PNG, GIF oder PDF mit entsprechender Änderung) im Hauptverzeichnis abgelegt sein. Ohne Selbstständigkeitserklärung wird keine Korrektur durchgeführt.

Die Erklärung muss die Selbstständigkeit wie auch die Prüfungsfähigkeit bestätigen. Ansonsten wird die Abgabe nicht gewertet. Sollte ein regelwidriger Täuschungsversuch aufgedeckt werden, so werden die Sanktionen der geltenden Prüfungsordnung angewandt.

Hinweis: Die beiden Textdateien sollten reinen Text im ASCII oder UTF-8 Format enthalten (Markdown wäre z.B. ggf. erlaubt; nutzen Sie dann die Endung `.md` statt `.txt`). Bitte keine Officedateien abgeben (da solche Programme auf den Testsystemen nicht installiert sein werden). Für die Bilddatei (Scan oder Foto der ausgefüllten Erklärung zur Prüfung) ist jedes gängige Bildformat erlaubt (bei anderen als den oben angegebenen trägt der/die Studierende das Risiko, dass es nicht gelesen werden könnte). Können Sie die Erklärung nicht ausdrucken, so ist auch eine handschriftliche Reproduktion erlaubt (es muss dann aber alles komplett abgeschrieben werden und sinnerhaltend formatiert werden, damit die Erklärung gültig wird).

Hinweise zur Bewertung allgemein (unvollständig):

Allgemeine Fehler

- Fehler im Framework, die sich in mehreren Aufgaben auswirken können als Folgefehler gewertet werden. Dies setzt in der Regel voraus, dass Kommentare im Programm die Abhängigkeiten klarmachen, oder dass die Readme-Datei explizit auf die Fehler hinweist.
- Speicherzugriffsverletzungen bei Operationen gelten als Fehler in der Implementation. Treten diese in exotischen Spezialfällen auf, so werden bis zu 20% der Punkte abgezogen, treten sie regelmäßig auf, 50%, und falls Sie immer auftreten gilt die Aufgabe als nicht gelöst (-100%). Das reine Abfangen via signal / SEH[6] gilt nicht als korrekte Lösung.
- „Speicherlöcher" werden auch als Fehler gewertet (Punktabzug halb so stark wie bei Zugriffsverletzungen), wenn diese wiederholt auftreten und damit die Funktion des Programms beeinträchtigen (Einmalig allokierten Speicher nicht händisch freizugeben, sondern automatisch bei Programmende durch das Betriebssystem freigeben zu lassen ist zwar schlechter Stil, aber im Sinne der Bewertung ok).

Übersetzung

- Sollte das Programm sich nicht mit den mitgelieferten Build-Files direkt übersetzen lassen, so muss zunächst versucht werden, den Fehler zu beheben. Liegt eine Inkompatibilität zwischen Qt-Versionen bzw. Interpretationen des C++-Standards durch den Compiler vor, oder ein Problem mit der Plattform, die nicht vom Studierenden zu verantworten ist, wie z.B. Anpassungen für Mac nach Unix sollte kein Mac bei der Korrektur zur Verfügung stehen, (oder ähnlich gelagerte Probleme), so werden keine Punkte abgezogen. Dies sollte eigentlich bei den gestellten Aufgaben nicht auftreten (da weder komplexe Qt oder C++ Features noch plattformspezifische Anpassungen benötigt werden).
- Sind minimale Änderungen nötig (z.B. Groß-/Kleinschreibung auf einem Unix-System), so wird 1 Punkt abgezogen. Probleme wie eine fehlende Datei oder fehlende Vorverarbeitungsschritte (moc/rcc o.ä.) werden mit 2-4 Punkten Abzug, je nach Aufwand der Änderungen bewertet.
- Ist die Struktur des Buildfiles ungeeignet, so gibt es 0/5 Punkten.
- Muss die Programmlogik selbst verändert werden, so gilt das Programm als „nicht übersetzbar". Je nach Schwere der Fehler werden 10 oder mehr Punkte abgezogen. Ausgenommen sind Änderungen von Programlogik, die lediglich zum Anpassen an eine bestimmte Plattform dienen (z.B. sollte spezieller Code in der Initialisierung entfernt werden müssen, der nur auf einem Mac funktioniert, aber die Korrektur geschieht auf einem anderen System). Gelingt eine Übersetzung gar nicht, ist eine Gesamtbewertung mit 0 Punkten möglich.

[6] Windows „Structured Execption Handling" – leitet OS-Fehler als Exception an die C++-Anwendung weiter.

Anhang:

Erklärung zur Take-Home Prüfung

Fachbereich: _____

Institut: _____

Name, Vorname: _____ (Matr.-Nr.: _____),

Die Take-Home-Prüfung im Fach „Einführung in die Softwareentwicklung“ (Dozent: M. Wand) findet als Projektarbeit vom 03.-09. August 2020 (ggf. zuzüglich Verlängerungen) statt.

[**WICHTIG**: Bitte bestätigen Sie die einzelnen Punkte durch Setzen eines Häkchens.]

Erklärung der Prüfungstauglichkeit

☐ Ich bin prüfungsfähig. Ich sehe mich derzeit physisch und psychisch dazu in der Lage, die Take-Home-Prüfung zu absolvieren.[a]

Erklärung der Selbständigkeit

gemäß § 3 Abs. 3 der Teil-Rahmenprüfungsordnung der Johannes Gutenberg-Universität Mainz für die Durchführung eines vorwiegend digitalen Semesters (Corona-Satzung) zur selbständigen und ordnungsgemäßen Teilnahme an einer Take-Home-Prüfung.

☐ Ich habe die vorliegende Take-Home-Prüfung selbstständig abgelegt und keine anderen als die erlaubten Quellen oder Hilfsmittel benutzt. Alle erlaubten Quellen (z.B. Literatur oder Internet), die im Sinne der am Anfang der Prüfungsaufgaben erklärten Regeln wesentlich zur Lösung beigetragen haben, habe ich in der Datei `Quellen.[txt|md]` oder `Readme.[txt|md]` in meiner Abgabe aufgeführt. (Die Dateiendungen txt/md bezeichnen hier die zur Wahl stehenden Dateiformate.)

☐ Diese Erklärung wurde mir zu Beginn der Prüfung zugänglich gemacht. Über die Folgen eines Täuschungsversuches wurde ich von meinem Prüfer / meiner Prüferin informiert.[b]

_____ _____ _____

(Ort) (Datum) (Unterschrift)

---

[a] Im Krankheitsfall besteht die Möglichkeit einen Rücktritt nach Ihrer einschlägigen Prüfungsordnung zu erklären. Dies muss unverzüglich geschehen. Bitte wenden Sie sich an das für Sie zuständige Prüfungsamt/Studienbüro. Sollten Sie sich trotz Krankheit der Prüfung unterziehen, führt dies zum Verlust des entsprechenden Rücktrittsrechts.

[b] Erweist sich die vorliegende Erklärung als unwahr oder liegt ein sonstiger Täuschungsversuch oder ein Ordnungsverstoß bei der Erbringung von Prüfungsleistungen vor, gelten die Regelungen der jeweiligen prüfungsrechtlichen Ordnung entsprechend.