Prüfung **Programmentwicklung 2**

Nachname	Vorname
Matrikelnummer	Studiengang

Hinweise:

- Bearbeitungszeit: 120 Minuten.
- Die Bearbeitung darf nur mit dokumentenechten Stiften erfolgen. Verwenden Sie keinen Bleistift. Verwenden Sie keinen Stift, der in Rot schreibt.
- Bei unterschiedlichen abgegebenen Lösungen für eine Aufgabe, wird die Aufgabe nicht bewertet. Antworten, die der Korrektor nicht lesen kann, werden nicht bewertet.
- Verwenden Sie keine Korrekturflüssigkeiten, *TippEx*, Tintenkiller oder Ähnliches. Nicht zu bewertende Lösungsteile sind durchzustreichen.
- Erlaubte Hilfsmittel: ein nicht-programmierbarer Taschenrechner sowie alle weiteren nicht-technischen Hilfsmittel.
- Mobiltelefone, Smartwatches und ähnliche Geräte sind ausgeschaltet in der Tasche zu verwahren. Die Verwendung eines solchen Gerätes wird als Täuschungsversuch gewertet.
- Sollte der vorgesehene Platz für die Lösung nicht ausreichen, verwenden Sie die Rückseiten des Bogens. Machen Sie eindeutig kenntlich, wo die Lösung weitergeht.

Durch Unterschrift bestätigen Sie, dass Sie die Hinweise zur Bearbeitung der Klausur gelesen und verstanden haben und sie als Bestandteile der Prüfungsbedingungen anerkennen.

Unterschrift

Aufgabe:	1	2	3	4	5	6	7	8	Σ	**Note**
Punkte:	9	16	25	20	10	20	20	20	140	
Erreicht:										

1. Prüfer: Prof. Dr. Michael Gref

2. Prüfer*in

1 Aufgabe (Fehlersuche) [9]

Gegeben sei der folgende Code für die Klassenhierarchie von Eingabegeräten, der einen oder mehrere Fehler beinhaltet, und die erwartete Ausgabe. Finden und korrigieren Sie den oder die Fehler direkt im Codeausschnitt oder als Freitext unter Angabe der entsprechenden Zeilennummern.

```
1  #include <iostream>
2  #include <string>
3
4  class Eingabegeraet {
5    private:
6    int id;
7    std::string bezeichnung;
8
9    public:
10   int get_id() { return id; };
11   Eingabegeraet (std::string bez) : id(0), bezeichnung(bez) {}
12
13   void increment_id(int offset) {
14     id + offset;
15   }
16 };
17
18 class Tastatur : public Eingabegeraet {
19   private:
20   Tastatur() : Eingabegeraet ("Tastatur") {}
21
22   void increment_id() {
23     id + 10;
24   }
25 };
26
27 class Maus: public Eingabegeraet {
28   private:
29   Maus() : Eingabegeraet ("Maus") {}
30
31   void increment_id() {
32     id + 20;
33   }
34 };
35
36 int main() {
37   Tastatur tastatur;
38   Maus maus;
39   tastatur.increment_id();
40   maus.increment_id();
41   std::cout << "Tastatur-ID: " << tastatur.get_id() << std::endl;
42   std::cout << "Maus-ID: " << maus.get_id() << std::endl;
```

```
43    return 0;
44 }
```

Erwartete Ausgabe:

```
Tastatur-ID: 10
Maus-ID: 20
```

2 Aufgabe (Programmverständnis) [16]

2.1 Teilaufgabe [8]

Gegeben sei das folgende C++-Programm:

```
#include <iostream>

class Value {
  protected:
  int _value;

  public:
  Value(int value) : _value(value) {
    std::cout << "K: " << _value << std::endl;
  }

  Value(const Value& m) : _value(m._value) {
    std::cout << "C: " << _value << std::endl;
  }

  ~Value() {
    std::cout << "D: " << _value << std::endl;
  }

  Value& operator++() {
    _value = _value + 1;
    std::cout << "I: " << _value << std::endl;
    return *this;
  }
};

int main() {
  Value m1(1);
  Value m2 = ++m1;
  Value m3(3);
  m3 = ++m2;
  ++m3;
  return 0;
}
```

Wie ist die vollständige Ausgabe in der Konsole?

2.2 Teilaufgabe [8]

Gegeben sei das folgende C++-Programm:

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

template<typename K, typename V>
void printMap(const std::map<K, V> &myMap) {
  for (const auto &pair: myMap) {
    std::cout << pair.first << ": " << pair.second << "\n";
  }
  std::cout << std::endl;
}

class C {
  public:
  int _x;
  C(int x) : _x(x) {};
};

std::ostream &operator<<(std::ostream &os, const C &c) {
  os << c._x;
  return os;
}

int main() {
  std::map<std::string, double> map1 =
      {{"Apfel",   0.2}, {"Kirsche", 0.05}, {"Banane",  0.3}};
  printMap(map1);

  std::map<C, double> map2 = {{C(2), 12.0}, {C(1), 42.0}};
  printMap(map2);

  return 0;
}
```

Hinweis: Laut Compiler ist der gegebene Code für map2 nicht funktional!
Teilaufgaben:

a) Wie ist die Ausgabe des Programms für map1, wenn wir die Zeilen zu map2 vorübergehend auskommentieren? [3]

b) Wie ist der Quell-Code zu ergänzen oder zu korrigieren, sodass das Programm auch für map2 funktional ist? [3]

c) Wie ist die Ausgabe des Programms für map2 nach der Korrektur? [2]

(Platz für Lösungen)

3 Aufgabe (Polymorphie) [25]

Ziel der Aufgabe ist die Entwicklung eines Systems zur Verwaltung verschiedener Arten von Events bzw. Veranstaltungsterminen. Das System soll einfach um neue Veranstaltungstypen erweitert werden können und unterschiedliche Teilnahme- und Preisgestaltungsstrategien ermöglichen. Dies sind die **Anforderungen**:

- Abstrakte Basisklasse `Event`:
  - Gemeinsame Eigenschaften für alle Events:
    `id (int), name (string), datum (string)`
  - Die ID soll für jedes Event einzigartig sein und automatisch beim Konstruktoraufruf gesetzt werden. Beim ersten erzeugten Objekt der abgeleiteten Klassen beginnt die ID mit 1 und wird bei jedem weiteren Objekt um jeweils 1 erhöht.
  - Eine dynamisch gebundene Methode `info()`, Rückgabetyp `string`, die Name und Datum des Event zurückliefert.
  - Eine abstrakte Methode `getPreis()`, die die Teilnahmegebühr basierend auf dem Eventtyp berechnet und zurückgibt.
- Abgeleitete Klassen:
  - `Webinar`: Zusätzliche Attribute:
    `thema (string), dauer (int)`
  - `Workshop`: Zusätzliche Attribute:
    `leiter (Person), teilnehmerLimit (int)`
  - `Konzert`: Zusätzliche Attribute:
    `kuenstler (Person), genre (string)`
- Eine Klasse `Person` zum Speichern und Zurückgeben des Namens der Person

Treffen Sie bei unvollständigen Informationen sinnvolle Annahmen und wählen Sie, falls notwendig, geeignete Beispielwerte.

Teilaufgaben *(Gerne für die Lösungen auch die Rückseite der vorherigen Seite (links) oder die Rückseite dieser Seite verwenden.)*

1. Skizzieren Sie ein vollständiges UML-Klassendiagramm unter Berücksichtigung der obigen Anforderungen. Berücksichtigen Sie alle getroffenen Spezifikationen, genannten Klassen, Methoden, Attribute und skizzieren alle Klassenbeziehungen. [10]
2. Geben Sie (ausschließlich) für die abstrakte Basisklasse und die Klasse für Konzert eine Implementierung in `C++` an, die die obigen Anforderungen realisiert. Die Methode `info()` soll in Konzert überschrieben werden und zusätzlich den Namen der Person sowie das Genre ausgeben. [10]

3. Schreiben Sie einen Komponententest (UnitTest), der die korrekte Realisierung der dynamischen Bindung für die Methode `info()` für Konzert sicherstellt, wenn über einen Zeiger vom Typ Event auf das Objekt zugegriffen wird. [5]

4 Aufgabe (Ein-/Ausgabe) [20]

In dieser Aufgabe sollen Sie eine Funktion in `C++` entwickeln, die aus einer Textdatei die Registrierungsdaten für verschiedene Seminare einer Konferenz einliest und eine Namensliste für alle Personen erstellt, die an einem spezifischen Seminar teilnehmen. Implementieren Sie die Funktion:

```
int printNameList(const string& filename, int seminarID)
```

Diese Funktion erhält als Argumente den Pfad zu einer Textdatei und die numerische ID eines Seminars. Jede Zeile in der Datei enthält die Registrierungsdaten einer Person für die Konferenz. Die Daten sind durch Kommas getrennt und folgen diesem Format: Name, Beruf, Liste von Seminar-IDs (ebenfalls mit Kommas getrennt). D.h. eine Person kann an mehreren Seminaren teilnehmen! Beispiel für den Inhalt der Datei:

```
Max Mustermann,Design,101
Erika Musterfrau,Entwicklung,102
Hanna Stark,Forschung,101,103
```

Beispiel: Wird die Funktion mit ID 101 aufgerufen soll die Ausgabe wie folgt aussehen:

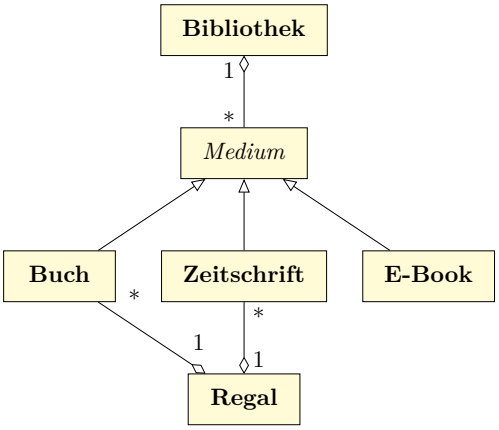
```
Teilnehmer des Seminars 101:
- Max Mustermann (Design)
- Hanna Stark (Forschung)
```

Rückgabe der Funktion ist die Anzahl an Teilnehmern für das Seminar. Sollte die Datei nicht geöffnet werden können, soll eine Fehlermeldung in die Konsole geschrieben werden und der Wert -1 zurückgegeben werden.

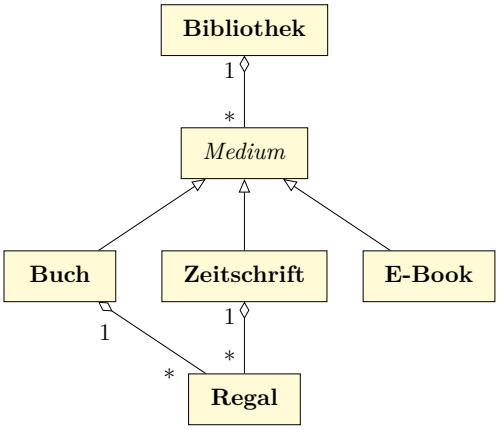
5 Aufgabe (Klassenbeziehungen) [10]

Begründen Sie kurz für jedes nachfolgende UML-Diagramm, ob eine Realisierung auf Basis des jeweiligen Diagramms geeignet ist, die Zusammenhänge der folgenden Aussage als Klassenbeziehung zu modellieren oder nicht.

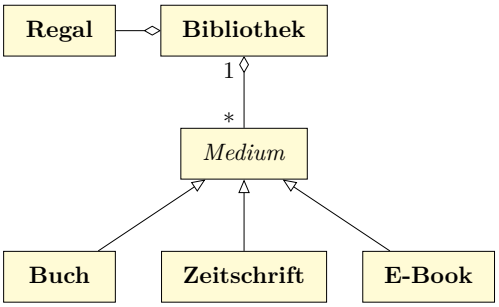
Aussage: Eine Hochschulbibliothek verfügt über eine Sammlung von Büchern, Zeitschriften und E-Books. Jedes physische Medium (Bücher und Zeitschriften) ist einem bestimmten Regal zugeordnet.



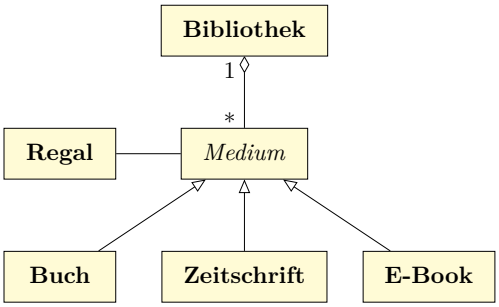
(a)



(b)



(c)



(d)

6 Aufgabe (Standard Template Library) [20]

Schreiben Sie eine `C++`-Funktion `void kostenUebersicht()`, die von Benutzern über die Konsole eingegebene Zahlungen erfasst und sie entsprechend ihrer Kategorie und des Datums organisiert. Die Eingabe erfolgt (wie im Beispiel) nicht unbedingt chronologisch. Nachdem die Nutzer die Eingabe mit dem Wort `ende` beenden, soll die Funktion eine strukturierte Zusammenfassung aller Zahlungen ausgeben. Die Zusammenfassung soll zuerst die Kategorie (alphabetisch sortiert) und pro Kategorie eine chronologische Aufstellung aller Ausgaben auflisten.

Sollte eine Eingabe erfolgen, die nicht dem erwarteten Format entspricht, soll eine Ausnahme des Typs `std::runtime_error` geworfen werden. Eine Ausnahmebehandlung ist nicht zu implementieren.

Beispiel Eingabe:

```
2024-07-02 Lebensmittel 100
2024-07-01 Lebensmittel 250
2024-07-01 Unterhaltung 150
2024-07-03 Bildung 300
ende
```

Erwartete Ausgabe:

```
Bildung:
2024-07-03: 300

Lebensmittel:
2024-07-01: 250
2024-07-02: 100

Unterhaltung:
2024-07-01: 150
```

Achten Sie darauf, alle notwendigen Header-Dateien anzugeben.

7 Aufgabe (Entwurfsmuster) [20]

Sie entwickeln eine Multimedia-Software, die es Nutzern ermöglicht, Audioaufzeichnungen zu bearbeiten und mit akustischen Effekten zu manipulieren (sogenanntes *Post-Processing*). Die Software soll vielfältige Effekte unterstützen, die auf die Audio-Tracks angewendet werden können. Beispiele für solche Effekte sind **Echo**, **Verstärkung** und **Rauschunterdrückung** (Denoise). Es können mehrere Effekte (auch mehrfach) verwendet werden und die Reihenfolge der Effekte ist relevant. Diese Effekte sollen bei Erzeugung eines Objekts hinzugefügt und bei Aufruf einer Methode `play()` entsprechend der Reihenfolge angewandt und abschließend wiedergegeben werden.

Das Ziel ist es, unter Einsatz objektorientierter Programmierung eine flexible und erweiterbare Softwarearchitektur zu entwerfen, sodass auch in Zukunft neue Effekte implementiert und hinzugefügt werden können, ohne die bestehenden Klassen zu ändern.

Teilaufgaben:

1. Welches der aus der Vorlesung bekannten Entwurfsmuster würde diese Anforderungen am besten erfüllen und warum? **Begründen** Sie, wie dieses Muster die in der Aufgabe geforderten Anforderungen ermöglicht! [6]

2. Skizzieren Sie ein UML-Klassendiagramm für Ihren Vorschlag und **erklären** Sie kurz(!) die Rolle jeder Klasse in Ihrem Diagramm. [8]
   (Lösung gerne wieder auf die Rückseite)

3. Geben Sie an, wie ein Objekt gemäß Ihres Entwurfs für einen AudioTrack erzeugt wird, wenn in folgender Reihenfolge die nachfolgenden Effekte hinzugefügt werden: Rauschunterdrückung, Verstärkung, Echo. [3]

4. Geben Sie die Definition der Methode `play()` für die Klasse an, die *Echo* realisiert. Die Klassendefinitionen aller Klassen (auch für Echo) können als vollständig bekannt und implementiert angesehen werden. Der Effekt kann mit der bereits implementierten Methode `applyEcho()` realisiert werden. [3]

8 Aufgabe (Verteilte Systeme) [20]

Gegeben ist im Folgenden eine unvollständige `C++`-Implementierung einer Socket-Wrapper-Klasse. Ihre Aufgabe ist die Implementierung der fehlenden Methoden.

Die geforderten Funktionssignaturen entnehmen Sie dem untenstehenden Quellcode. Verwenden Sie für Strings ausschließlich Teile der `C++`-Standardbibliothek!

Die send-Methode soll einen übergebenen String über den Socket der Klasse verschicken. Der String soll dabei durch die Zeichenkette '$$$' terminiert werden. Verschickte Nachrichten sollen zusätzlich über die Konsole ausgegeben werden. Achten Sie auf eine angemessene Fehlerbehandlung inklusive Konsolenausgaben.

Die recv-Methode soll Strings über den Socket der Klasse, in 256 Byte Blöcken, empfangen. Eingehende Blöcke sollen, bis zum Empfang des Terminierungsstrings, zusammengesetzt werden. Leere Blöcke werden dabei ignoriert. Der empfangene String soll ohne Terminierungssymbole zurückgegeben werden. Achten Sie auf eine angemessene Fehlerbehandlung inklusive Konsolenausgaben.

Teilaufgaben

1. Dem Entwickler der Socket-Bibliothek ist ein kleiner Fehler bei der Socket-Konfiguration unterlaufen. Finden und korrigieren Sie den Fehler. [5]
2. Implementieren Sie eine send-Methode, welche den oben beschriebenen Anforderungen entspricht. [5]
3. Implementieren Sie eine recv-Methode, welche den oben beschriebenen Anforderungen entspricht. [10]

Hinweis: Sie müssen nicht den bestehenden Quellcode abschreiben.

socket.hpp:

```
1  #include <string>
2  #include <netinet/in.h>
3
4  class Socket {
5    private:
6    sockaddr_in _address;
7    int _socket;
8
9    public:
10   Socket(std::string ip, int port);
11   Socket(int socket);
12
13   void send(std::string msg);
```

```
14   std::string recv(void);
15   void close(void);
16 };
```

socket.cpp:

```
1  #include <string>
2  #include <cerrno>
3  #include <cstring>
4  #include <sys/socket.h>
5  #include <netinet/in.h>
6  #include <arpa/inet.h>
7  #include <unistd.h>
8  #include <iostream>
9  #include "socket.hpp"
10
11 Socket::Socket(std::string ip, int port) {
12   _socket = ::socket(AF_INET, SOCK_STREAM, 0);
13   if (_socket == -1)
14   throw std::runtime_error(strerror(errno));
15
16   _address.sin_family = AF_INET;
17   _address.sin_addr.s_addr = ::inet_addr(ip.c_str());
18   _address.sin_port = port;
19   int len = sizeof(_address);
20
21   int rc = ::connect(_socket, (struct sockaddr *) &_address, len);
22   if (rc == -1)
23   throw std::runtime_error(strerror(errno));
24 }
25
26 Socket::Socket(int socket) {
27   _socket = socket;
28 }
29
30 void Socket::send(std::string msg) {
31   // Ihr Code
32 }
33
34 std::string Socket::recv(void) {
35   // Ihr Code
36 }
37
38 void Socket::close(void) {
39   ::close(_socket);
40 }
```

(Platz für Lösungen)