Musterklausur im FACH ***OOSWE***

Name, Vorname: …….

Studiengang/Semester: ……

Bearbeitungshinweise

1. Tragen Sie auf jeder Seite in der Kopfzeile Ihre Matrikelnummer ein.

2. Der Aufgabensatz (Deckblatt und Anhang), der aus 6 Seiten besteht (Seite 1 bis 6) ist auf Vollständigkeit zu überprüfen.

3. Der Aufgabensatz ist mit den Lösungsblättern abzugeben.

4. Lösungen auf selbst mitgebrachten Lösungsblättern werden nicht ausgewertet. Verwenden Sie die Ihnen ausgeteilten Lösungsblätter und tragen Sie auch dort Ihre Matrikelnummer ein.

5. Bei Rechenaufgaben muss der Lösungsweg ersichtlich und lesbar sein, sonst erfolgt keine Bewertung der Aufgabe oder des Aufgabenteils.

6. Die Bearbeitungszeit beträgt 60 Minuten.

7. Es wird hiermit darauf hingewiesen, dass vom Prüfungsamt nicht vorher geprüft wurde, ob Sie das Recht bzw. die Pflicht zur Teilnahme an dieser Klausur haben. Die Teilnahme erfolgt auf eigene Gefahr, gleichzeitig bekundet die Teilnahme die Zustimmung zu diesem Passus.

8. Hilfsmittel:
   - Coding Styleguide (ist selbst mitzubringen) – nur Markierungen mit einem Marker sind erlaubt (keine eigenen Notizen)
   - Taschenrechner sind **nicht** erlaubt – auch **keine** Smartphones

9. **Die Nichteinhaltung des Coding Styleguides führt zu Punktabzug**

10. Bewertung:
    Gesamtpunktzahl: 65 Punkte (10% Überhang)
    Note 1,0: 60 Punkte Note 4,0: 30 Punkte

Aufgabe	**1**	**2**	**3**	**4**	**SUMME**
Punkte	**20**	**20**	**10**	**15**	**65**
Erreichte Punkte					

Aufgabe 1: (20 Punkte)

1.1 Die Verwendung von _____ gehört auch zum „C++ Knigge“. Insbesondere ist dies bei Libs (statisch und dynamisch) sehr wichtig! (1 Punkt)

1.2 Nennen Sie zwei Vorteile von Referenzen gegenüber Zeigern: (2 Punkte)

-
-

1.3 Gegeben sei die Klasse A und die beiden Deklarationen einer überladenen Methode:

```
void foo(A a);
void foo(A& a);
```

Warum gibt es einen Compilerfehler wenn foo (siehe unten) aufgerufen wird? (2 Punkte)

```
int main(void)
{
   A a;
   foo(a);
   return 0;
}
```

1.4 Gegeben sei die folgende Anweisung:

```
uint32_t* pu32Val = malloc(100 * sizeof(uint32_t));
```

Warum gibt es in C++ einen Compilierfehler und in C nicht? (2 Punkte)

1.5 Wird C++ Code schon in der Headerdatei implementiert, so ergibt sich implizites _____ (1 Punkt).

1.6 Zwei Threads inkrementieren eine Variable

```
u32Global++;
```

Der Maschinencode sieht dabei wie folgt aus:

```
mov   eax,dword ptr [u32Global (01823D0h)]
add   eax,1
mov   dword ptr [u32Global (01823D0h)],eax
```

Welcher Fehler kann dabei entstehen? (2 Punkte)

1.7 Wie ist das assert-Makro für NDebug definiert? (2 Punkte)

```
#define assert(expression)
```

1.8 Wie muss Produktivcode in C++ aufgebaut sein, um diesen mit GoogleTest testen zu können? Zeigen Sie dies anhand eines Blockdiagramms auf! (2 Punkte)

1.9 Gegeben sei der folgende C++ Code

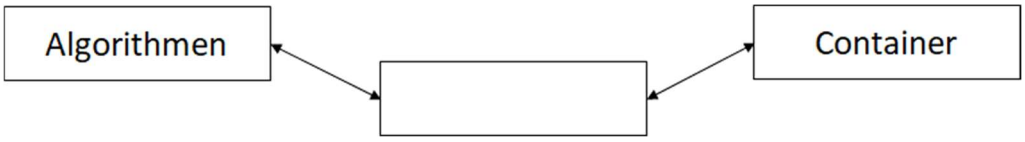
```
class BaseClass
{
public:
   BaseClass(uint32_t u32V) : u32V_(u32V) {}
   virtual void vfuncA() = 0;
   virtual void vfuncB() = 0;
protected:
   uint32_t u32V_;
};

class DerivedClass : public BaseClass
{
public:
   DerivedClass(uint32_t u32V) : BaseClass(u32V) {}
   void vfuncA() override {}
   void vfuncB() override {}
};

DerivedClass* pdc = (DerivedClass*) new DerivedClass{2U};
```

Zeichnen Sie rechts den belegten Speicher des instanziierten Objektes in den Speicherbereich (Zelle entspricht Byte, oben hohe Adressen) ein.
Beschreiben Sie auch kurz, was in dem Speicher steht.
Platform x86, Microsoft Visual Studio (4 Punkte)

1.10 Welcher Begriff fehlt im mittleren Block (Hinweis: STL)? (1 Punkt)



1.11 Gegeben sei ein Vector vectors32 für int32_t. Der folgende Aufruf des find-Algorithmus hat eine Komplexität von O(____). (1 Punkt)

```
std::vector<int32_t>::iterator itres = std::find(vectors32.begin(), vectors32.end(), 42);
```

Aufgabe 2: (20 Punkte)

Es soll ein Dateisystem mit **Verzeichnissen (Dictories)** und **Dateien (Files)** als Design Pattern in C++ realisiert werden. Ein Verzeichnis kann weitere (Unterverzeichnisse) Verzeichnisse sowie Dateien enthalten. Als Attribut soll jede der nicht abstrakten Klassen einen std::string strName_ vererbt bekommen.

2.1 Zeichnen Sie das Klassendiagramm des Design Patterns ohne Methoden aber mit Attributen (5 Punkte). Abstrakte Klassen bitte mit {abstract} kennzeichnen.

2.2 Begründen Sie die Wahl des STL-Containers für obiges Design Pattern! (2 Punkt)

2.3 Implementieren Sie die Klasse Verzeichnis mit allen für das Design Pattern notwendigen Methoden. Verwenden Sie auch einen passenden Smart Pointer! Konstruktor und Destruktor nicht vergessen. Wählen Sie für die Methoden des STL-Containers selbstsprechende Namen. Diesen müssen nicht mit den konkreten Namen übereinstimmen. Sie finden daher im Anhang auch keine Kurzreferenz! (13 Punkte)

Lösung auf dem Lösungsbogen

Aufgabe 3: (10 Punkte)

Implementieren Sie ein Template mit dem Namen Pair, welches zwei Objekte o1 und o2 (können von unterschiedlichen Klassen O1 und O2 sein) enthält. Die Objekte o1 und o2 sind dem Konstruktor per Initialization list zu übergeben.
Das Template soll den Subtraktionsoperator als Objektmethode überschreiben, welcher o1a – o1b und o2a – o2b berechnet (a steht für den ersten Operanden, b für den zweiten Operanden). Die Subtraktion für O1 und O2 sei vorhanden (O1 könnte z.B. uint32_t und O2 könnte f64_t sein). Der Subtraktionsoperator soll das Originalobjekt überschreiben und verkettbar sein.

```
template <                                    > // 1P
class Pair
{
public:
   Pair(                          ) :                                    //Initialization list 2P
   { }

   Pair   operator-(Pair p)          // 1P
   {
                                     // 4 P









                                     // 1 P
   }

private: // 1 P



};
```

Aufgabe 4: **15 Punkte**

4.1 Welche drei C++ Schlüsselwörter gibt es zur Kapselung der Attribute? (1 Punkt)

4.2 Die „kleine Schwester" einer Klasse (class) ist ein _____(Keyword). (1 Punkt)
Hinweis: Im gesamten Buch von Lopinoso kommt das Keyword class nie vor!

4.3 Eine Klassenmethode erkennt man am vorangestellten Keyword _____. (1 Punkt)

4.4 Laut Coding Styleguide lassen sich Klassenvariablen und Objektvariablen (Attribute) wie folgt unterscheiden: (1 Punkt)

4.5 Sie wollen verhindern, dass andere Mitarbeiter in einer Objektmethode die Objektvariablen (Attribute) beschreiben. Wie könnten Sie dies im Code verhindern? (1 Punkt)

4.6 Gegeben sei die Klasse Car. Diese enthält noch den Standard-Konstruktor (default). Instanziieren Sie einmal statisch und einmal dynamisch ein Objekt dieser Klasse mit „Brace yourself". (2 Punkte)

4.7. Gegeben seien die Klassen B und C.
Implementieren Sie den Kopierkonstruktor für tiefe und flache Kopien sowie den CopyMove-Konstruktor. (6 Punkte)

```
class B
{
private:
   C* pC_;
};
```

Kopierkonstruktor (flache Kopie)

Kopierkonstuktor (tiefe Kopie)

Move-Konstruktor

4.8 Welcher der obigen Konstruktoren aus 4.7 erfordert welche Voraussetzung der Klasse C. Wenn diese nicht vorhanden ist, funktioniert der Konstruktor auch nicht (Compilierfehler) (2 Punkte)

Konstruktor: _____ Voraussetzung: C muss _____