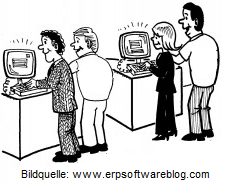
**Und**

**Modul 320**

***(Mini-Projektarbeit)***



**Ziel der Arbeit:**

Der vermittelte Unterrichtsstoff des Moduls 320 (Objektorientiert Programmieren) soll in einer individuellen Umsetzungsaufgabe so vertieft werden, dass die Lernenden mit ihrer Arbeit zeigen, dass die Inhalte der Modulidentifikation verstanden wurden. Diese Umsetzungsaufgabe wird mit einer Note (die zweite Im Modul 320) bewertet.

**Arbeitsform:**

Einzelarbeit oder Partnerarbeit (max. 2). Bei einem Team muss aber die Arbeitsweise klar deklariert werden:

|  |  |  |
| --- | --- | --- |
| Arbeitsweise | Beschreibung | Auswirkungen auf die Note |
| as a team with different tasks | Jedes Mitglied hat die eigenen Aufgaben. Diese Tasks werden zu Beginn besprochen und geplant (ist in der Dokumentation ersichtlich) | Anhand der Taskplanung werde ich abschätzen, ob eine gleiche oder unterschiedliche Notenverteilung erfolgen sollte um (Umfang und Komplexität) gerecht zu werden. |
| Pair development | Alles wird gemeinsam gemacht, beide wechseln sich ab und leisten den gleichen Beitrag bezüglich Umfang und Kompexität | Gleiche Note für beide |
| knowledge transfer | Eine Person ist sehr kompetent und die zweite Person möchte davon profitieren (geschult werden) und übernimmt darum eher niederschwellige Arbeiten. | Typischerweise wird die schulende Person mehr leisten. Die zweite Person kann aber viel profitieren. Die Note wird bei der schulenden Person etewas höher sein. Die Note der geschulten Person wird durch Verständnisfragen bestummen (Lernfortschritt) |

**Zur Verfügung stehender Zeitrahmen (~6 Lektionen):**

Es stehen ihnen an folgenden Tagen entsprechende Zeit dazu zur Verfügung:

|  |  |  |  |
| --- | --- | --- | --- |
| **Projektwoche** | **Datum** | **M320** | **Ziel** |
| 01 | 15.12.2023 | 2 Lekt | Teambildung, Taskplanung, Ideenfindung |
| 02 | 22.12.2023 | 2 Lekt | Arbeiten am Projekt |
| 03 | 12.01.2024 | 2 Lekt | Arbeiten am Projekt, Abgabe um 23.59 Uhr |

Abgabetermin ist: 12.01.2024 / 23.59 Uhr. Dazu wird eine Aufgabe in Ihrem Team existieren via der sie ihre Arbeit (z.B. den Git-Link) zustellen. Die Arbeit muss zwingend via Teams-Aufgabe abgegeben werden.

**Umfang der Projektarbeit:**

Zeigen sie mit ihrer Arbeit, dass sie die Handlungsziele des Moduls 320 gem. Modulbeschrieb verstanden haben und über die Kompetenz des Moduls verfügen.

Generell gilt: **Qualität ist wichtiger als Quantität!**

**Programmcode**

Was wird von Ihnen erwartet? Sie müssen mit Hilfe einer objektorientierten Programmiersprache (C#, Java, Phyton, etc.) ein Programm umsetzen.

* Der Programmcode soll sauber gekappselt sein (Membervariablen sind private oder protected und der Zugriff erfolgt über public Methoden.
* Das Programm muss mind. eine Vererbungsbeziehungen enthalten und mind. ein Interface welches von mind. einer Klasse implementiert wird.
* Die Vererbung und der Einsatz des Interface müssen einen Sinn/Nutzen ergeben
* Das Programm muss mind. eine Assoziationsbeziehung vom Typ «1 -n» enthalten. Ein Objekt muss also eine Membervariable enthalten, die ein dynamisches Array referenziert. Das Array wiederum referenziert div. Andere Objekte.
* Die Konstruktoren werden mit base/super Aufrufen sinnvoll/effizient aufgerufen
* Sie erstellen mindestens einen Kompnententest (Unittest) der eine unit of code überprüft.

**Dokumentation:**

* Sie machen auf Papier vor der Umsetzung einen Entwurf eines groben Klassendiagramms (welche Klassen wie miteinander in Beziehung stehen). Dieses Dokument müssen sie einscannen und mit der Arbeit abgeben.
* Sie machen eine Paketplanung, die zeigt, welche Arbeitspakete existieren, welchen Aufwand sie für die Erledigung eines Paketes schätzen und wer welches Paket bis wann erledigen muss. Später stellen sie den geschätzten Aufwand und den effektiven Aufwand gegenüber.
* Sie generieren mit einer UML-Software aus dem Programmcode ein detailliertes UML-Klassendiagramm und stellen mind. einen Aspekt des Programms als UML-Sequenzdiagramm dar.
* Mittels Screenrecording präserntieren sie Ihre Arbeit in einem 4-8 minütigen Video. Sie zeigen zuerst die Idee ihrer Applikation und den geplanten groben Aufbau (grobes Klassendiagramm). Zeigen sie danach, wie sie die Beziehungen umgesetzt haben. Zeigen Sie anhand der Ausführung eines Unittests, dass Ihre Test funktionieren. Am Schluss zeigen sie ihr Sequenzdiagramm und springen im Debugmodus Schritt für Schritt durch die Methodenaufrufe des Sequenzdiagramms.