INTERNATIONALE HOCHSCHULE

PORTFOLIO

Aufgabenstellung zum Kurs: Objektorientierte und funktionale Programmierung mit Python (DLBDSOOFPP01_D)

INHALTSVERZEICHNIS

1. AUFGABENSTELLUNG

Im Rahmen dieses Portfoliokurses muss die folgende Aufgabenstellung bearbeitet werden.

1.1. Aufgabe: Entwicklung eines Dashboards für Dein Studium

Im Laufe Deines Studiums sind viele Kurse und deren Prüfungsleistungen zu absolvieren. Damit Du den Überblick nicht verlierst, sollst Du ein **Dashboard für Dein Studium entwerfen** und **prototypisch** mit **Python** umsetzen. Das Dashboard soll Dir einen schnellen Überblick über Deinen Studienfortschritt geben. Als Ausgangspunkt kannst Du Dir anschauen, welche Informationen zu Deinem Studium in myCampus zur Verfügung stehen. Dort findest Du zum Beispiel eine Übersicht über Deine aktuellen Kurse oder Deine Prüfungsergebnisse.

Es geht aber nicht nur um ein tolles Dashboard, das entstehen soll, sondern eine besondere Rolle spielt die **objektorientierte Modellierung des Dashboards** und wie die **objektorientierten Konzepte mit der Programmiersprache Python umgesetzt werden**.

1.1.1. Konzeptionsphase

Die Konzeptionsphase ist der wichtigste Teil im Portfolio. Alles was hier erarbeitet wird, beeinflusst direkt die Ergebnisse der weiteren Phasen. Folgende Inhalte sind in der Konzeptionsphase zu erarbeiten:

1. **Festlegen** der **Ziele**, die mit dem Dashboard überwacht werden sollen. Ein konkretes Ziel könnte zum Beispiel sein, dass Du Dein Studium in 3 Jahren abschließen möchtest, oder dass Du Dein Studium mit einem Notendurchschnitt von 2,0 abschließen möchtest. Du kannst auch auf die zwei genannten Ziele fokussieren. Für den Fall, dass Du noch weitere Ziele/Kennzahlen überwachen möchtest, kannst Du diese gerne ergänzen. Beachte aber, dass es nicht zu viele werden, da das Dashboard übersichtlich bleiben soll.

2. **Festlegen** der angezeigten **Inhalte** auf dem **Dashboard**. Was soll auf dem Dashboard genau wie dargestellt werden? Ein Dashboard muss alles Wesentliche beinhalten, um die aktuelle Zielerreichung schnell erkennen zu können. Achte darauf, dass Dein Dashboard übersichtlich bleibt. Das Ergebnis kann mit einem beliebigen Zeichenprogramm angefertigt werden. Es ist aber auch möglich, eine sorgfältig von Hand erstellte Skizze einzuscannen. Besonders relevant für die Bewertung ist, ob mit Deiner Dashboard-Skizze, die unter Punkt 1 festgelegten Ziele gut überwacht werden können.

3. **Erstellung** eines **UML-Klassendiagramm**. Das Klassendiagramm in dieser frühen Konzeptionsphase soll nur die sogenannten Entity-Klassen, zu denen später auch etwas gespeichert werden soll, beinhalten. Ein Studiengang ist zum Beispiel in Semester untergliedert[1]. Einem Semester sind Module zugeordnet. Zu einem Modul

[1] Siehe hierzu zum Beispiel https://www.iu.de/fernstudium/softwareentwicklung-bachelor/ unter Inhalte.

gibt es eine Prüfungsleistung. Aus diesem Zusammenhang aus der Realität lassen sich im Rahmen der Modellierung die Entity-Klassen Studiengang, Semester, Modul und Prüfungsleistung ableiten. Die gefundenen Klassen und deren Beziehungen sind in einem UML-Klassendiagramm[1] darzustellen. Besondere Bedeutung für die Bewertung hat die korrekte Verwendung der objektorientierten Konzepte[2] wie zum Beispiel Klasse, Abstraktion, Vererbung, Komposition oder Aggregation. Das Klassendiagramm ist auf die korrekte und sinnvolle Verwendung dieser objektorientierten Konstrukte hin zu untersuchen und die getroffenen Modellierungsentscheidungen sind zu begründen und zu diskutieren. Bitte beachte, dass das erstellte Klassendiagramm vollständig unabhängig von einer konkreten Programmiersprache ist! Die Umsetzung des Klassendiagramms und der objektorientierten Konzepte mit Hilfe von Python wird in der Erarbeitungs- und Reflexionsphase untersucht!

4. **Machbarkeitsüberprüfung** und **Erproben** von **Umsetzungsmöglichkeiten mit Python**. Dein Dashboard soll in den folgenden beiden Phasen mit Hilfe von Python prototypisch umgesetzt werden. Damit es dann nicht zu unerwarteten Überraschungen kommt, sollst Du Dich bereits in dieser Phase mit folgenden Fragen beschäftigen:
   - Welche Python Entwicklungsumgebung[3] und welche Python Bibliotheken sollen warum zum Einsatz kommen?
   - Soll die Datenspeicherung in Dateien oder in einer Datenbank erfolgen?
   - Wie soll der Quellcode verwaltet werden?
   - Soll die Interaktion mit dem Benutzer über die Kommandozeile oder über eine grafische Bedienoberfläche erfolgen?

Für die Beantwortung der genannten Fragen musst Du kleine Testprogramme in Python schreiben, um sicherzugehen, dass Deine Überlegungen umsetzbar sind. Sollen die Daten zum Beispiel später in einer Datei gespeichert werden, musst Du ein kleines Python-Programm schreiben, in welchem Du die Werte von Variablen in eine Datei schreibst und später wieder aus der Datei mit Hilfe Deines Programmes einliest. Alle Entscheidungen sind entsprechend zu begründen. Für die spätere Bewertung ist besonders relevant, dass die Entscheidungen gut begründet und nachvollziehbar sind.

Vielleicht fragst Du Dich, was denn die prototypische Umsetzung des Dashboards in Python bedeutet. Prototypische Umsetzung bedeutet, dass das am Ende erstellte Python-Programm aufzeigen soll, dass Dein Konzept in Python umsetzbar ist. Das Ergebnis des Portfolios muss also keine App sein, die an einen Endkunden ausgeliefert werden kann. Deshalb ist es auch nicht zwingend erforderlich (wenngleich möglich), dass Dein Python-Programm eine grafische Bedienoberfläche besitzt – es können auch die einzelnen Funktionen beispielsweise über die Kommandozeile angesprochen werden.

Die Ergebnisse der Konzeptionsphase (Punkt 1 bis 4) sind in einem **Konzeptdokument** festzuhalten. Das Konzeptdokument muss als PDF in PebblePad als Ergebnis der Phase 1 hochgeladen werden und darf maximal 5 Seiten umfassen. Der Inhalt des Konzeptdokumentes besteht also aus

- den Zielen,
- der Dashboard-Skizze inklusive Beschreibung,
- dem UML-Klassendiagramm inklusive kritischer Diskussion der Modellierungsentscheidungen,
- einer Aufstellung welche Programmierumgebungen, Werkzeuge, etc. in welchen Versionen für die spätere Umsetzung warum ausgewählt wurden.

Das Textfeld in der PebblePad Vorlage kann freigelassen werden.

[1] Zum Zeichnen des UML-Diagramms solltest Du einen UML-Editor verwenden.
[2] Du solltest Dich deshalb auch besonders sorgfältig mit den objektorientierten Konzepten wie Klasse, Abstraktion, Vererbung, usw. vertraut machen.
[3] Achte darauf, eine aktuelle Python-Version zu verwenden!

Während der Erarbeitung Deines Portfolios gibt es im Rahmen der Online-Tutorien die Möglichkeit, Deine Ergebnisse der Konzeptionsphase zu diskutieren und sich Feedback einzuholen. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Verbesserungen vorzunehmen. Erst danach sollen die Ergebnisse der ersten Phase zur Bewertung abgegeben werden.** Danach erfolgt ein abschließendes Feedback durch die Tutorin/den Tutor und die Arbeit in der zweiten Phase kann beginnen.

1.1.2. Erarbeitungs-/Reflexionsphase

Folgende Inhalte sind in der Erarbeitungs-/Reflexionsphase zu erarbeiten:

1. **Untersuchung** der **Umsetzung objektorientierter Konzepte in Python**. Hierzu sollst Du in übersichtlichen kleinen Testprogrammen die objektorientierten Konzepte, die für Dein Klassendiagramm aus Phase 1 relevant sind, untersuchen. Zum Beispiel kannst Du untersuchen, wie eine Klasse Prüfungsleistung in Python programmiert wird oder wie eine Vererbungsbeziehung programmiert wird. Besonders wichtig ist hierbei, wie die objektorientierten Konzepte in Python umzusetzen sind. Welche Besonderheiten gibt es? Lässt sich Dein Klassendiagramm aus der Konzeptionsphase in Python ohne Veränderung so umsetzen oder möchtest Du noch Änderungen durchführen?

2. **Erstellen der Gesamtarchitektur** für Deinen Prototypen. Nach den Erkenntnissen aus der vorangegangenen Untersuchung sollst Du Deine Gesamtarchitektur für Dein Dashboard erstellen. Gibt es noch zusätzliche Klassen, die benötigt werden? Wenn ja, welche und welche Aufgabe erfüllen diese? Zum Beispiel werden häufig zusätzliche Klassen für die Interaktion mit dem/der Benutzer:in oder für das Speichern der Daten verwendet. Als Ergebnis der Gesamtarchitektur entsteht ein UML-Klassendiagramm, das alle Python-Klassen Deines späteren Dashboard-Prototypen und deren Beziehungen zueinander zeigt.

Die Ergebnisse der Erarbeitungs- und Reflexionsphase (Punkt 1 und 2) sind in einem **Reflexions- und Entwurfsdokument** festzuhalten. Das Reflexions- und Entwurfsdokument muss als PDF in PebblePad als Ergebnis der Phase 2 hochgeladen werden und darf maximal 5 Seiten umfassen. Der Inhalt des Reflexions- und Entwurfsdokumentes besteht damit aus:

- Einer Diskussion der Umsetzungsmöglichkeit der objektorientierten Konzepte in Python (Text, kein Programmcode, einzelne Programmcodeausschnitte sind zur Erläuterung erlaubt – müssen aber explizit in der Diskussion aufgegriffen werden).
- Die Gesamtarchitektur als UML-Klassendiagramm inklusive einer textuellen Erläuterung.

Das Textfeld in der PebblePad Vorlage kann freigelassen werden.

Während der Erarbeitung Deines Portfolios gibt es im Rahmen der Online-Tutorien die Möglichkeit, Deine Ergebnisse dieser Phase zu diskutieren und sich Feedback einzuholen. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Verbesserungen vorzunehmen. Erst danach sollen die Ergebnisse in der zweiten Phase zur Bewertung abgegeben werden**. Nach dem folgenden abschließenden Feedback Durch die Tutorin/den Tutor wird in der dritten Phase an der Fertigstellung des Dashboard-Prototypen weitergearbeitet.

1.1.3. Finalisierungsphase

Folgende Inhalte sind in der Finalisierungsphase zu erarbeiten:

1. **Implementierung** des **Dashboard-Prototypen in Python** anhand der Gesamtarchitektur aus Phase 2. Der Programmcode ist im Quelltext sauber zu dokumentieren.
2. **Bereitstellen** des **Dashboard-Prototypen** über einen **GitHub-Link**. Falls Du noch kein GitHub-Konto besitzt, musst Du eines erstellen. In GitHub erstellst Du ein Repository und lädst Deinen gesamten Quellcode in dieses Repository hoch.
3. **Erstellung** einer **Installationsanleitung**. Dein Programmcode soll schnell durch die Tutorin/den Tutor analysiert und „getestet" werden können. Je einfacher die Installationsanleitung ist und je schneller es Deiner Tutorin/Deinem Tutor gelingt den Programmcode „zum Laufen zu bringen", desto besser. In dieser Installationsanleitung muss sich auch der GitHub-Link auf Dein Repository befinden. Teste Deine Installationsanleitung auch unbedingt auf einem aktuellen Windows-Betriebssystem.
4. **Erstellung** eines **Projektverzeichnisses**. Alle Ergebnisse des Portfolios aus allen 3 Phasen (inklusive des Programmcodes auf GitHub) sollen in einem Projektverzeichnis strukturiert abgelegt werden. Der Inhalt des gesamten Projektverzeichnisses ist am Ende der Phase 3 als Zip-Datei abzugeben. Siehe hierzu auch die Ordnerstruktur für die Finalisierungsphase in Kapitel 4.2.
5. **Erstellung** eines **Abstracts** mit einem Umfang von **1 bis 2 Seiten**. Stelle Dir vor, Du schreibst zum Projektende einen Bericht an Deinen Vorgesetzten. Was wurde umgesetzt? Wie war die Herangehensweise? Was lief gut? Was lief schlecht? Welche Erkenntnisse gab es? Auf was bist Du besonders stolz? Wie können Deine Ergebnisse weiterverwendet werden?

Die **Ergebnisse der Finalisierungsphase** (Punkt 1 bis Punkt 5) sind wie folgt final abzugeben:

- Inhalt des **kompletten Projektverzeichnisses** (inklusive des Programmcodes mit integrierter Dokumentation, der sich auch auf GitHub befindet) als zip-Datei.
- **Installationsanleitung inklusive GitHub-Link** im Umfang von maximal einer Seite **als PDF**.
- Das **Abstract** im Umfang von 1 – 2 Seiten **als PDF**.

Auch in der Finalisierungsphase gibt es im Rahmen der Online-Tutorien die Möglichkeit, sich ausreichend Rückmeldung, Tipps und Hinweise zu holen, **bevor** das fertige Produkt final abgegeben wird. **Es wird empfohlen, diese Kanäle zu nutzen, um Fehler zu vermeiden und Verbesserungen vorzunehmen**.

2. BETREUUNGSPROZESS

Bei der Betreuung der Portfolios stehen grundsätzlich mehrere Kanäle offen. Die jeweilige Inanspruchnahme liegt dabei im eigenen Verantwortungsbereich. Die eigenständige Erarbeitung eines Produktes und die Befüllung der jeweiligen Portfolioteile ist dabei Teil der zu erbringenden Prüfungsleistung und fließt in die Gesamtbewertung mit ein.

Zum einen sieht die tutorielle Betreuung Feedbackschleifen zu den einzureichenden Portfolioteilen im Rahmen der Konzeptions- sowie der Erarbeitungs- und Reflexionsphase vor. Das Feedback erfolgt im Rahmen einer Einreichung des jeweiligen Portfolioteils. Des Weiteren werden regelmäßige Online-Tutorien angeboten, in denen Gelegenheit besteht, mit der Tutorin/dem Tutor Fragen zur Bearbeitung des Portfolios zu besprechen. Die Tutorin/der Tutor steht zusätzlich für fachliche Rücksprachen sowie für formale und allgemeine Fragen zum Vorgehen bei der Portfoliobearbeitung zur Verfügung.

Technische Fragen zur Nutzung von PebblePad sind per Mail an das Prüfungsamt zu richten.

3. ZUSATZINFORMATIONEN ZUR BEWERTUNG

In die Bewertung des Portfolios fließen die folgenden Kriterien mit dem jeweils angegebenen Prozentsatz ein:

Bewertungskriterien	Erläuterungen	Gewichtung
Problemabgrenzung/Zielsetzung	*Erfassung des Problems *Klare Problemabgrenzung/Zielsetzung *Nachvollziehbares Konzept	10%
Methodik/Idee/Vorgehen	*Angemessener Transfer von Theorien/Modellen *Klare Angaben zur gewählten Methodik/zur gewählten Idee/zum gewählten Vorgehen	20%
Qualität der Umsetzung	*Qualität der Umsetzung und Dokumentation	40%
Kreativität/Richtigkeit	*Kreativität des Lösungsansatzes *Umgesetzte Lösung erfüllt angestrebte Zielsetzung	20%
Formale Anforderungen	*Einhaltung der formalen Vorgaben.	10%

Bei der Konzeption und Erstellung des Portfolios sollten die genannten Bewertungskriterien einschließlich der folgenden Erläuterungen berücksichtigt werden.

Problemabgrenzung/Zielsetzung: Sind die Ziele, die mit dem Dashboard überwacht werden sollen, klar formuliert? Ist die Dashboard-Skizze geeignet, die definierten Ziele zu überwachen?

Methodik/Idee/Vorgehen: Ist die Machbarkeitsüberprüfung sinnvoll begründet und nachvollziehbar? Sind die Werkzeuge gut gewählt und werden diese richtig und sinnvoll eingesetzt?

Qualität der Umsetzung: Sind die Klassendiagramme logisch richtig und UML-konform? Konnte das Klassendiagramm sinnvoll in Python-Code transferiert werden? Sind die Beschreibungen logisch richtig und nachvollziehbar? Sind die Kommentare im Programmcode sinnvoll? Ist die Installationsanleitung intuitiv und kann der Python-Code mit geringem Aufwand ausgeführt werden?

Kreativität/Richtigkeit: Wie kreativ ist das entwickelte Dashboard und der umgesetzte Prototyp? Gibt es besondere Aspekte in der Gestaltung, Umsetzung, Modellierung oder Werkzeugverwendung?

Formale Anforderungen: Sind die formalen Vorgaben (siehe Kapitel 4) erfüllt?

4. FORMALIA UND VORGABEN ZUR ABGABE

4.1. Bestandteile der Prüfungsleistung

Im Folgenden befindet sich eine Übersicht der Prüfungsleistung Portfolio mit seinen einzelnen Phasen, einzureichenden Einzelleistungen und Feedbackrunden im Überblick. Für die Erarbeitung der Portfolioteile im Rahmen der Prüfungsleistung wird eine Vorlage in PebblePad zur Verfügung gestellt. Die Vorlage ist Bestandteil dieser Prüfungsleistung.

Phase	Zwischenergebnis	Einzureichende Leistung
Konzeptionsphase	Portfolioteil 1	• Konzeptdokument mit den Inhalten wie in Kapitel 1.1.1 beschrieben als PDF. Maximal 5 Seiten.
		Feedback
Erarbeitungsphase/ Reflexionsphase	Portfolioteil 2	• Reflexions- und Entwurfsdokument wie in Kapitel 1.1.2 beschrieben als PDF. Maximal 5 Seiten.
		Feedback
Finalisierungsphase	Portfolioteil 3	• Ergebnis aus Phase 1 als PDF • Ergebnis aus Phase 2 als PDF • Komplettes Projektverzeichnis inklusive dem kompletten Programmcode und integrierter Programmcode-Dokumentation als zip-Datei • 1-2-seitiges Abstract als PDF • Installationsanleitung mit GitHub-Link auf Dein Repository als PDF. Maximal 1 Seite.
		Feedback + Note

4.2. Formalia zur Abgabe digitaler Dateien

Konzeptionsphase

Empfohlene Hilfsmittel/Software zur Bearbeitung	Textverarbeitungswerkzeug z. B. Word, UML-Editor, Zeichenprogramm
Zugelassene Dateiformate	PDF
Dateigröße	möglichst gering
Weitere Formalien und Parameter	**Die prüfungsleistungsrelevante Abgabedatei auf PebblePad ist wie folgt zu benennen:** Nachname-Vorname_Matrikelnummer_Kurs _Phase1.pdf Beispiel: Mustermann-Max_12345678_ Python _Phase1.pdf Textfeld bei der Abgabe kann leer gelassen werden. Das PDF muss angehängt werden.

Erarbeitungs-/Reflexionsphase

Empfohlene Hilfsmittel/Software zur Bearbeitung	Textverarbeitungswerkzeug z. B. Word, UML-Editor, Python Entwicklungswerkzeuge und Bibliotheken
Zugelassene Dateiformate	PDF
Dateigröße	möglichst gering
Weitere Formalien und Parameter	**Die prüfungsleistungsrelevante Abgabedatei auf PebblePad ist wie folgt zu benennen:** Nachname-Vorname_Matrikelnummer_Kurs _Phase2.pdf Beispiel: Mustermann-Max_12345678_ Python _Phase2.pdf Textfeld bei der Abgabe kann leer gelassen werden. Das PDF muss angehängt werden.

Finalisierungsphase

Empfohlene Hilfsmittel/Software zur Bearbeitung	Textverarbeitungswerkzeug z. B. Word, Python Entwicklungswerkzeuge und Bibliotheken
Zugelassene Dateiformate	PDF: Für alle Dokumente mit Beschreibungen und/oder Grafiken. Python-spezifische Dateiformate: Für alle Programmcode zugehörigen Dateien. ZIP: Für die Abgabedatei, welche die gesamten Ergebnisse enthält.
Dateigröße	möglichst gering
Weitere Formalien und Parameter	**WICHTIG** ist das Einfügen eines eigens für die Abgabe erstellten zip-Ordners. In diesem Ordner befinden sich verteilt auf Unterverzeichnisse alle Dateien. **Die Ordnerstruktur muss wie folgt aussehen:** Name Hauptverzeichnis (Benennung zip-Ordner): Nachname-Vorname_Matrikelnummer_Python • Name von Unterverzeichnis 1: Phase1 • Name von Unterverzeichnis 2: Phase2 • Name von Unterverzeichnis 3: Phase3 **Die Zip-Datei hat damit folgenden Namen:** Nachname-Vorname_Matrikelnummer_Python.zip **Die 1seitige Installationsanleitung muss folgendermaßen benannt werden:** Nachname-Vorname_Matrikelnummer_Kurs_Phase3_Installationsanleitung.pdf Beispiel: Mustermann-Max_12345678_ Python _Phase3_Installationsanleitung.pdf. **Das 1-2seitige Abstract muss folgendermaßen benannt werden:** Nachname-Vorname_Matrikelnummer_Kurs_Phase3_Abstract.pdf Beispiel: Mustermann-Max_12345678_ Python _Phase3_Abstract.pdf **Abgabe auf PebblePad:** Laden Sie das Ergebnis aus Phase 1 im Abschnitt Zwischenabgabe Phase 1 hoch. Laden Sie das Ergebnis aus Phase 2 im Abschnitt Zwischenabgabe Phase 2 hoch. Laden Sie den Zip-Ordner im Abschnitt Ressourcen hoch. Laden Sie das Abstract im Abschnitt Abstract hoch. Laden Sie die Installationsanleitung im Abschnitt Finales Produkt hoch.

4.3. Formalia für das Abstract

Umfang	1-2 Seiten Textteil
Papierformat	DIN A4
Seitenränder	Oben und unten 2cm; links 2cm; rechts 2cm
Schrifttyp	Allgemeiner Text – Arial 11Pkt; Überschriften – 12Pkt, Blocksatz
Zeilenabstand	1,5
Satz	Blocksatz und Silbentrennung
Fußnoten	Arial 10Pkt, Blocksatz
Absätze	Nach gedanklicher Gliederung – 6Pkt Abstand nach Zeilenumbruch
Eidesstattliche Erklärung	Die Abgabe der Eidesstattlichen Erklärung erfolgt in elektronischer Form über myCampus. Davor ist keine Einreichung der Prüfungsleistung möglich. Bitte beachtet hierzu die Anleitung für das Einreichen eines Portfolios in myCampus.

Bei Fragen zur Abgabe des Portfolios wende Dich bitte per Mail an das Prüfungsamt.

Beachte bitte zusätzlich die Nutzungsanleitung zu PebblePad & Atlas!

Viel Erfolg beim Erstellen des Portfolios!