**Advanced workbook**

Aufgabenstellung zum Kurs:

DLBCSDSJCL02\_D – Datenstruktur und Java-Klassenbibliothek

**Bitte überprüfe die Gültigkeit dieser Workbookaufgaben im Kurs auf myCampus.**

**Hinweis zum Urheberrecht und zur Plagiatsprüfung:**

**Es wird darauf hingewiesen, dass der IU Internationale Hochschule GmbH das Urheberrecht der Prüfungsaufgaben/Aufgabenstellungen obliegt. Einer Veröffentlichung der Aufgabenstellungen auf Drittplattformen wird ausdrücklich widersprochen. Im Falle einer Zuwiderhandlung stehen der Hochschule u.a. Unterlassungsansprüche zu. Zudem weisen wir darauf hin, dass jede eingereichte schriftliche Ausarbeitung mittels einer Plagiatssoftware überprüft wird. Wir empfehlen daher auch, keinesfalls ausgearbeitete Lösungen zu teilen, da dies den Verdacht eines Plagiates begründen kann.**

****Aufgabenstellung 1:****

**Überlege Dir drei Anwendungsbeispiele, in welchen die Dokumentation von Code (d. h. das Versehen von Code mit Kommentaren) in einer Forschungsgruppe einer Universität sinnvoll sein kann. Gehe dabei insbesondere auf zumindest zwei Anwendungsbeispiele ein, bei denen der Code nicht nur von Dir selbst verstanden werden soll.**

****Aufgabenstellung 2:****

**Erläutere den Unterschied der Verwendung des „java.util.ArrayList“-, des „java.util.TreeSet“- und des „java.util.TreeMap“-Frameworks am Beispiel eines von Dir gewählten Use Cases. Warum eignet sich die Verwendung des jeweiligen Frameworks gerade besonders gut für den von Dir gewählten Use Case?**

****Aufgabenstellung 3:****

**Erläutere das Konzept von Annotationen mittels eines selbst gewählten Code-Beispiels. Gehe dabei auf die beiden geläufigsten Annotationen @override sowie @depricated ein.**

****Aufgabenstellung 4:****

Angenommen, Du entwickelst eine Texteditor-Anwendung in Java und möchtest eine "Undo-Funktion" implementieren, mit der Benutzer ihre letzten Aktionen rückgängig machen können. Beschreibe, wie Du die Stack-Datenstruktur in Java verwenden könntest, um diese Undo-Funktion zu implementieren. Erkläre, wie der Stack funktioniert und wie er verwendet werden kann, um die Benutzeraktionen zu speichern und rückgängig zu machen. Diskutiere dabei auch die Schritte, die erforderlich sind, um die Undo-Funktion zu implementieren, einschließlich des Hinzufügens von Aktionen zum Stack, des Entfernens von Aktionen und der Durchführung des rückgängig gemachten Vorgangs. Gib Beispiele für die Verwendung des Stacks, um Benutzeraktionen zu verfolgen und die Undo-Funktion zu ermöglichen an. Diskutiere auch mögliche Herausforderungen bei der Implementierung der Undo-Funktion und stelle Lösungen vor, um diese Herausforderungen zu bewältigen. Bewerte abschließend die Vor- und Nachteile der Verwendung der Stack-Datenstruktur für die Implementierung der Undo-Funktion in Bezug auf Speicherbedarf und Ausführungsgeschwindigkeit.

****Aufgabenstellung 5:****

Schreibe eine Java-Klasse namens "Person", die zwei private Instanzvariablen enthält: "name" vom Typ String und "alter" vom Typ int. Die Klasse sollte einen Konstruktor haben, der den Namen und das Alter der Person als Parameter akzeptiert und die entsprechenden Instanzvariablen initialisiert.

Implementiere nun die equals()-Methode für die Person-Klasse. Die equals()-Methode sollte eine andere Person als Parameter akzeptieren und überprüfen, ob der Name und das Alter beider Personen übereinstimmen. Verwende dazu den ==-Operator, um das Alter zu vergleichen, und die equals()-Methode der String-Klasse, um die Namen zu vergleichen.

Schreibe anschließend eine main-Methode, um die Funktionalität der equals()-Methode zu testen. Erstelle zwei Person-Objekte mit demselben Namen und Alter und überprüfe, ob sie gleich sind, indem Du die equals()-Methode aufrufst. Gib entsprechende Ausgaben aus, um zu zeigen, ob die Personen gleich sind oder nicht.

****Aufgabenstellung 6:****

Beschreibe die Funktionalität und die Verwendung der LinkedHashMap-Klasse in Java. Erkläre die Unterschiede zwischen LinkedHashMap und HashMap und diskutiere die Vor- und Nachteile der Verwendung von LinkedHashMap. Gib Beispiele an, um den Einsatz von LinkedHashMap in verschiedenen Szenarien zu verdeutlichen und erläutere, warum diese Szenarien von der Verwendung von LinkedHashMap profitieren würden.