**Basic Java Elements**

**Aufgabe 1 – Java Reserved Words**

1. Was sind reservierte Wörter in Java (Java Reserved Words) und warum sind sie wichtig?
2. Nenne zehn Beispiele für reservierte Wörter in Java.
   1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   4. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   5. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   6. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   7. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   8. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   9. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
   10. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Erkläre den Unterschied zwischen einem reservierten Wort und einem Bezeichner (Identifier).

**Aufgabe 2 – Packages in Java**

1. Was sind Pakete (Packages) in Java und wofür werden sie verwendet?
2. Wie definiert man ein Paket in einer Java-Datei?
3. Was ist der Unterschied zwischen einem Standardimport und einem statischen Import? Gib ein Beispiel für jeden.
4. Erstelle ein kleines Java-Projekt, das die Nutzung von Paketen und Importen demonstriert:
   * Erstelle ein Paket namens com.example.utils.
   * Definiere in diesem Paket eine Klasse *MathUtils* mit einer Methode public static int add(int a, int b).
   * Erstelle ein weiteres Paket namens com.example.main.
   * Definiere in diesem Paket eine Klasse *Main*, die die Methode add aus der Klasse *MathUtils* verwendet.
   * Implementiere eine statische Methode print aus der System-Klasse, um das Ergebnis der add-Methode auszugeben.

**Aufgabe 3 – Aufbau einer Java-Quelldatei**

Skizziere und beschreibe den Aufbau einer Java-Quelldatei mit min. einer Klasse. Kennzeichne dabei alle optionalen und nicht optionalen Komponenten. Kennzeichne und begründe die Stellen, an denen bestimmte Formen der Kommentare (JavaDoc, Inline-Kommentare,…) angebracht wären.

**Aufgabe 4 – Kompilierung von Programmen I**

*Hinweis: Du kannst alternativ deine Lösung aus Aufgabe 2 erweitern und anpassen.*

1. **Erstelle die folgenden Verzeichnisstrukturen:**
   * com/example/util
   * com/example/app
2. **Erstelle die Klasse** com.example.util.MathUtils **in der Datei** MathUtils.java:
3. **Erstelle die Klasse** com.example.app.Main **in der Datei** Main.java:
4. **Kompilierung und Ausführung:**

**Schritt-für-Schritt-Anleitung:**

1. **Kompilieren der *MathUtils* Klasse:** Öffne ein Terminal oder eine Eingabeaufforderung und navigiere zum Verzeichnis, in dem sich die Datei MathUtils.java befindet. Führe den folgenden Befehl aus:

javac com/example/util/MathUtils.java

1. **Kompilieren der *Main* Klasse:** Navigiere zum Verzeichnis, in dem sich die Datei Main.java befindet, und führe den folgenden Befehl aus:

javac -cp . com/example/app/Main.java

1. **Ausführen des Programms:** Nachdem beide Klassen erfolgreich kompiliert wurden, kannst du das Programm ausführen, indem du den folgenden Befehl ausführst:

java -cp . com.example.app.Main

1. Welche Ausgabe erzeugt dein Programm?

**Aufgabe 5 – Kompilierung von Programmen II**

Erstelle eine Klasse *Person* und eine Klasse *Message*.

1. Kompiliere beide Klassen, um die .class-Dateien zu erstellen:

javac Person.java Message.java

1. Verwende den Befehl javap -d auf den erstellten .class-Dateien, um die Dekompilierungsinformationen anzuzeigen:

javap -d Person.class Message.class

**Aufgabe 6 – Standard-Packages in Java**

Ergänze den Package-Namen und gebe min. drei Beispielklassen an, die in diesem Package liegen.

Packages: *java.util, java.sql, java.lang, java.net, java.io, java.awt, java.nio, javax.swing*

|  |  |  |
| --- | --- | --- |
| Package | Erklärung | Beispiele |
|  | Grundlegende Klassen und Schnittstellen. Enthält grundlegende Java-Klassen, die automatisch in jedes Java-Programm importiert werden. |  |
|  | Dienstprogramme wie Sammlungen, Kalender und Zufall. Enthält nützliche Dienstprogrammklassen für Datenstrukturen, Datum/Uhrzeit und mehr. |  |
|  | Klassen für Eingabe und Ausgabe (I/O). Bietet Klassen zur Eingabe und Ausgabe von Daten, z.B. Dateioperationen. |  |
|  | Klassen für nicht blockierende I/O-Operationen. Ermöglicht nicht blockierende I/O-Operationen und verbesserte Dateioperationen. |  |
|  | Klassen für Netzwerkprogrammierung. Bietet Klassen zur Implementierung von Netzwerkverbindungen und -kommunikation. |  |
|  | Klassen für Datenbankzugriff und SQL-Operationen. Ermöglichen die Verbindung und Interaktion mit Datenbanken unter Verwendung von SQL. |  |
|  | Klassen für das Erstellen von Benutzeroberflächen. Enthält Klassen zur Erstellung von grafischen Benutzeroberflächen (GUIs) mit dem Abstract Window Toolkit (AWT). |  |
|  | Erweiterte Komponenten für GUI-Entwicklung. Bietet eine Reihe von fortgeschrittenen GUI-Komponenten, die flexibler und leistungsfähiger sind als AWT-Komponenten. |  |

**Aufgabe 7 – Object, System und Random-Klassen in Java**

Ergänze zu den Klassen eine kurze Erläuterung:

java.lang.Object:

java.lang.System:

java.util.Random: