Java Data Types

Aufgabe 1 – Primitive Datentypen

a) Ergänze die passenden primitiven Datentypen und ihre zugehörigen Wrapper-Klassen.

Primitiver Datentyp	Wrapper-Klasse	Minimumwert	Maximumwert	Standardinitialisierung
		-128	127	0
		-32,768	32,767	0
		-2,147,483,648	2,147,483,647	0
		-9,223,372,036,854,775,808	9,223,372,036,854,775,807	0L
		1.4E-45	3.4028235E38	0.0f
		4.9E-324	1.7976931348623157E308	0.0d
		\u0000 (null)	\uFFFF	'\u0000' (null)
		false	true	false

b) Markiere die Integer Datentypen rot, die Gleitkommazahlen grün, die character Datentypen blau und die Wahrheitswerte gelb.

Aufgabe 2 - Bezeichner

Ein Bezeichner (Identifier) in der Programmierung ist ein Name, der verwendet wird, um Variablen, Funktionen, Methoden, Klassen oder andere benannte Elemente zu kennzeichnen. Bezeichner helfen dabei, den Code lesbar und verständlich zu machen, indem sie den einzelnen Elementen aussagekräftige Namen geben.

In Java gibt es bestimmte Regeln für die Benennung von Bezeichnern:

1. _
2. _
3. _
4. _

Aufgabensammlung: Wiederholung 811

Aufgabe 3 – Initialisierung und Referenzieren von Variablen

Folgendes Programm hast du gegeben:

```
public class Person {
    String name;
    int age;
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public void printInfo() {
        System.out.println("Name: " + name + ", Alter: " + age);
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        // Primitive Variablen
        int a = 5;
        int b = a; // Kopieren des Werts
        b = 10;
        System.out.println("a: " + a); // a bleibt unverändert
        // Referenzvariablen
        Person person1 = new Person("Alice", 25);
        Person person2 = person1; // Kopieren der Referenz
        person2.name = "Bob";
        person1.printInfo(); // Name wird als "Bob" ausgegeben, da person1 und person2 auf dasselbe Objekt zeigen
        // Ein weiteres Beispiel für Referenzvariablen
        changeName(person1);
        person1.printInfo(); // Name wird als "Charlie" ausgegeben, da das Objekt innerhalb der Methode geändert wurde
    }
    public static void changeName(Person person) {
        person.name = "Charlie";
    }
}
```

Führe das Programm aus und analysiere die Ausgabe:

Erkläre, warum sich der Wert der primitiven Variable a nicht ändert, obwohl b geändert wurde.

Erkläre, warum sich der Name von person1 ändert, wenn person2 geändert wird.

Erkläre, warum der Name von person1 innerhalb der Methode changeName geändert wird.

Aufgabe 4 – Pass by Value und Pass by Reference

a) Erkläre jeweils in drei bis fünf Sätzen, was die Konzepte Pass by Value und Pass by Reference bedeuten.
b) Ergänze Vor- und Nachtteile der jeweiligen Konzepte.
c) Beschreibe jeweils exemplarisch Situationen, in denen du das Konzept bevorzugen würdest und warum.

Pass by Value:

Erklärung:

Vorteile und Nachteile:

Anwendungen:

Pass by Reference:

Erklärung:

Vorteile und Nachteile:

Anwendungen:

Aufgabe 5 – Initialisierung von Variablen

Rechercheauftrag: Wann ist eine Variable initialisiert?

Aufgabe 6 - Casten

In Java gibt es die Möglichkeit zwischen verschiedenen Datentypen zu wechseln. Erkläre die nachfolgenden Begriffe und gebe jeweils zwei Beispiele an.

Implizites Casten:

Explizites Casten:

Upcasting:

Downcasting: