Kontrollstrukturen in Java

In Java werden Kontrollstrukturen verwendet, um den Fluss eines Programms basierend auf bestimmten Bedingungen oder Wiederholungen zu steuern. Zu den wichtigsten Kontrollstrukturen zählen die bedingten Anweisungen (`if`, `else`, `switch`) und Schleifen (`for`, `while`, `do-while`). Diese ermöglichen die Ausführung von Code abhängig von Bedingungen oder wiederholt bis eine Bedingung erfüllt ist.

Bedingte Anweisungen

1. `if` und `else`-Anweisung

Die `if`-Anweisung führt Code nur aus, wenn eine Bedingung wahr (`true`) ist. Mit `else` kann man Alternativen angeben, falls die Bedingung falsch (`false`) ist.

```
int age = 20;

if (age >= 18) {
    System.out.println("Du bist volljährig.");
} else {
    System.out.println("Du bist minderjährig.");
}
```

2. `else if`-Anweisung

Die `else if`-Anweisung erlaubt es, mehrere Bedingungen nacheinander zu prüfen, bevor ein `else`-Block als Fallback ausgeführt wird.

```
int score = 85;

if (score >= 90) {
    System.out.println("Note: A");
} else if (score >= 80) {
    System.out.println("Note: B");
} else if (score >= 70) {
    System.out.println("Note: C");
} else {
    System.out.println("Nicht bestanden");
}
```

3. `switch`-Anweisung

Die `switch`-Anweisung ist nützlich, wenn man den Wert einer Variablen gegen mehrere mögliche Fälle (`case`) prüfen möchte.

```java
int day = 3;
String dayName;

switch (day) {
    case 1:
        dayName = "Montag";
        break;
    case 2:
        dayName = "Dienstag";
        break;
    case 3:
        dayName = "Mittwoch";
        break;
    case 4:
        dayName = "Donnerstag";
        break;
    case 5:
        dayName = "Freitag";
        break;
    case 6:
        dayName = "Samstag";
        break;
    case 7:
        dayName = "Sonntag";
        break;
    default:
        dayName = "Ungültiger Tag";
        break;
}

System.out.println("Heute ist " + dayName);
```

Schleifen

Schleifen sind Strukturen, die es ermöglichen, Anweisungen wiederholt auszuführen, solange eine bestimmte Bedingung erfüllt ist.

1. `for`-Schleife

Eine `for`-Schleife ist ideal für bekannte Wiederholungen, bei denen die Anzahl der Iterationen im Voraus festgelegt ist.

```java
for (int i = 0; i < 5; i++) {
    System.out.println("Durchlauf Nummer: " + i);
}
```

2. `while`-Schleife

Die `while`-Schleife wiederholt eine Anweisung, solange eine Bedingung wahr (`true`) ist. Diese Schleife eignet sich, wenn die Anzahl der Durchläufe nicht im Voraus bekannt ist.

```
int count = 0;

while (count < 5) {
    System.out.println("Durchlauf Nummer: " + count);
    count++;
}
```

3. `do-while`-Schleife

Die `do-while`-Schleife ähnelt der `while`-Schleife, jedoch wird der Codeblock mindestens einmal ausgeführt, da die Bedingung erst am Ende geprüft wird.

```
int count = 0;

do {
    System.out.println("Durchlauf Nummer: " + count);
    count++;
} while (count < 5);
```

Erweiterte Kontrollstrukturen

1. `break`-Anweisung

`break` wird verwendet, um eine Schleife oder einen `switch`-Block vorzeitig zu beenden.

```
for (int i = 0; i < 10; i++) {
    if (i == 5) {
        break; // Schleife wird abgebrochen, wenn i 5 erreicht
    }
    System.out.println("i: " + i);
}
```

2. `continue`-Anweisung

`continue` überspringt die aktuelle Iteration und fährt mit der nächsten Schleifeniteration fort.

```
for (int i = 0; i < 10; i++) {
    if (i % 2 == 0) {
        continue; // Gerade Zahlen werden übersprungen
    }
```

```
        System.out.println("Ungerade Zahl: " + i);
    }
```

3. Verschachtelte Schleifen

Schleifen können auch ineinander verschachtelt werden, um mehrdimensionale Strukturen (wie Tabellen) zu durchlaufen.

```
for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= 3; j++) {
        System.out.print(i * j + " ");
    }
    System.out.println(); // Zeilenumbruch nach jeder inneren Schleife
}
```

Kontrollstrukturen Zusammenfassung

Kontrollstruktur	Erklärung	Beispiel
`if`	Führt einen Codeblock aus, wenn eine Bedingung wahr ist.	`if (age >= 18) { System.out.println("Volljährig"); }`
`else if`	Prüft eine zusätzliche Bedingung, wenn die vorherige `if`-Bedingung falsch ist.	`else if (score >= 80) { System.out.println("Note: B"); }`
`else`	Fallback-Codeblock, der ausgeführt wird, wenn keine vorherige Bedingung erfüllt ist.	`else { System.out.println("Nicht bestanden"); }`
`switch`	Überprüft eine Variable gegen mehrere mögliche Werte (`case`-Blöcke).	`switch (day) { case 1: ... break; }`
`for`	Führt eine festgelegte Anzahl von Wiederholungen aus, typischerweise bei bekannten Durchläufen.	`for (int i = 0; i < 5; i++) { ... }`
`while`	Wiederholt eine Anweisung, solange eine Bedingung wahr ist.	`while (count < 5) { ... }`
`do-while`	Ähnlich wie `while`, führt aber den Codeblock mindestens einmal aus.	`do { ... } while (count < 5);`
`break`	Beendet eine Schleife oder einen `switch`-Block vorzeitig.	`if (i == 5) { break; }`
`continue`	Überspringt die aktuelle Schleifeniteration und fährt mit der nächsten fort.	`if (i % 2 == 0) { continue; }`

Kontrollstruktur	Erklärung	Beispiel
Verschachtelte Schleifen	Führt Schleifen in einer weiteren Schleife aus, z. B. für tabellenartige Strukturen.	`for (int i = 1; i <= 3; i++) { for (int j = 1; j <= 3; j++) { ... } }`

Diese Kontrollstrukturen sind die Basis für die Steuerung des Programmflusses in Java und ermöglichen das Erstellen flexibler und leistungsfähiger Anwendungen.