

FMUD

Termin 2

FMuD

Ergänzung Schleifen

Konstrukturen

Vererbung

Ergänzung Schleifen

break und continue

break;

Beendet Schleife komplett

continue;

Beendet aktuellen Schleifendurchlauf

Ergänzung Schleifen

break und continue

in einer while-Schleife

```
while(Bedingung)
{
    ...
    continue;
    ...
    break;
    ...
}
```

The diagram illustrates the execution flow of a while loop. A green line starts at the 'continue;' statement, moves right, then up, then left, and finally down to the start of the loop body, indicating that the current iteration is skipped and the next iteration begins. A brown line starts at the 'break;' statement, moves right, then up, then left, and finally down to the end of the loop, indicating that the loop is terminated.

Ergänzung Schleifen

break und continue

in einer do-while-Schleife

```
do  
{
```

```
...
```

```
continue;
```

```
...
```

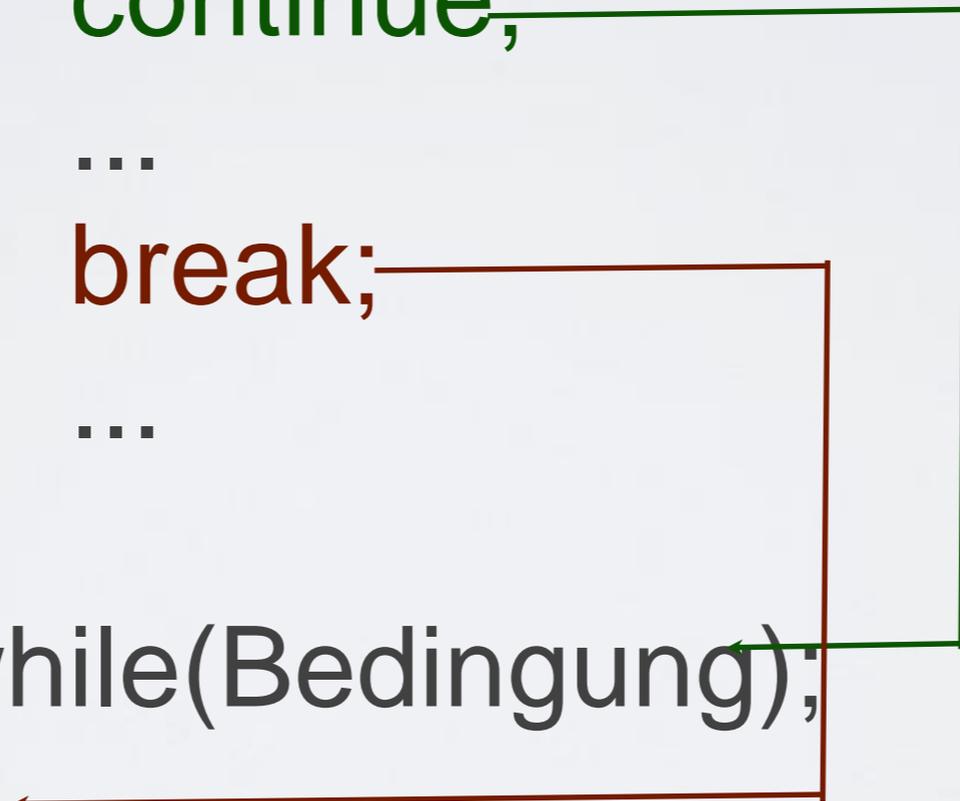
```
break;
```

```
...
```

```
}
```

```
while(Bedingung);
```

```
...
```



Ergänzung Schleifen

break und continue

in einer for-Schleife

```
for(Initialisierung, Bedingung, Update)
{
    ...
    continue;
    ...
    break;
    ...
}
...
```

The diagram illustrates the execution flow of a for loop. A green box highlights the `continue;` statement and its path back to the `Bedingung` parameter of the `for` loop. A red box highlights the `break;` statement and its path out of the loop to the next line of code.

Aufgabe 6

Geben Sie mittels einer endlosen while-Schleife,
alle positiven ganzen Zahlen aus.

Die Zahl 10 soll jedoch nicht ausgegeben werden und
die Schleife soll abgebrochen werden, wenn der Wert
> 20 ist.

Realisieren Sie diese Aufgabe mit Hilfe von break und
continue.

Konstruktoren

Ein Konstruktor wird aufgerufen, wenn ein neues Objekt einer Klasse erzeugt wird.

Name des Konstruktor \Leftrightarrow Name der Klasse

Konstruktoren können überladen werden

Aufgabe 7

Erstellen Sie neben ihrer Main-Klasse eine Klasse „Konto“.

Die Klasse soll die Attribute „Konto-Nummer“ und „Kontostand“ enthalten. (Attribute sollen private sein.)

Es soll sowohl ein parameterloser Konstruktor, so wie ein Konstruktor, dem Parameter „mitegegeben“ werden, erstellt werden.

Erstellen Sie außerdem eine Methode „print“, die die gespeicherten Attribute ausgibt.

Aufgabe 7a

Erstellen Sie ein Objekt der Klasse Konto, belegen Sie die Attribute mit Werten und geben Sie diese mittels der print()-Methode aus.

Vererbung

Um von einer Klasse zu erben wird in der erbenden Klasse „extends **Klassenname-der-vererbenden-Klasse**“

nach den Klassennamen gesetzt
Der Konstruktor der vererbenden Klasse kann mit
„super();“
aufgerufen werden.

Methoden der vererbenden Klasse können mit
„super.**Methodenname**();“
aufgerufen werden.

Aufgabe 8

Leiten Sie von der Klasse „Konto“ eine Klasse „Sparkonto“ ab. Die Klasse Sparkonto soll ein zusätzliches Attribut „Zinssatz“ enthalten.

Die Klasse Sparkonto soll ebenfalls zwei

Konstruktoren (einen mit, einen ohne Parameter), sowie eine Methode „print“, die die gespeicherten Attribute ausgibt, besitzen.

Vergessen Sie die Attribute der Oberklasse nicht.

Aufgabe 8a

Erzeugen Sie ein Objekt der Klasse
Sparkonto,
belegen Sie die Attribute mit Werten
und geben Sie diese mittels der print()-
Methode aus.