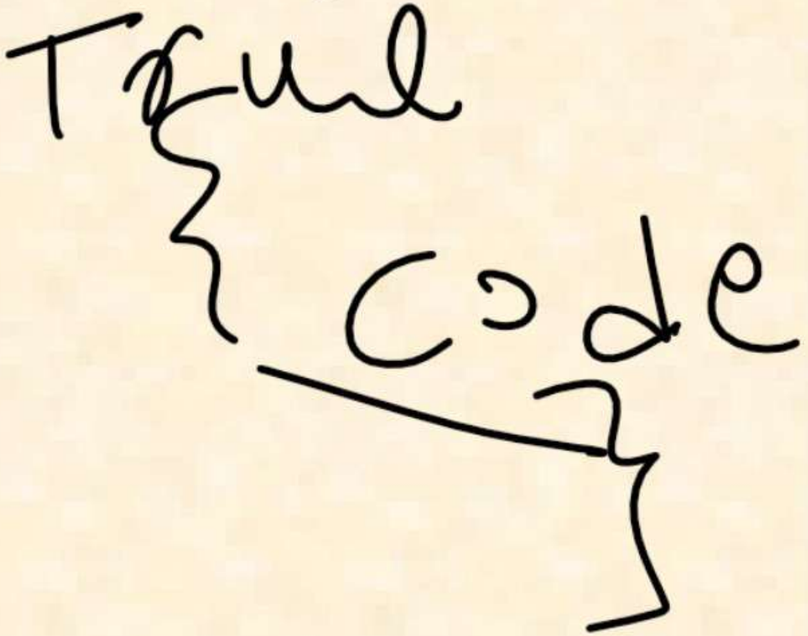
Conditional Statements

Conditional Statements

Conditional statements in Java are used to make decisions in a program. They allow your code to execute **different actions based on whether a condition is true or false**.

In real life, we make decisions like:
If it's raining, take an umbrella; otherwise, don't.

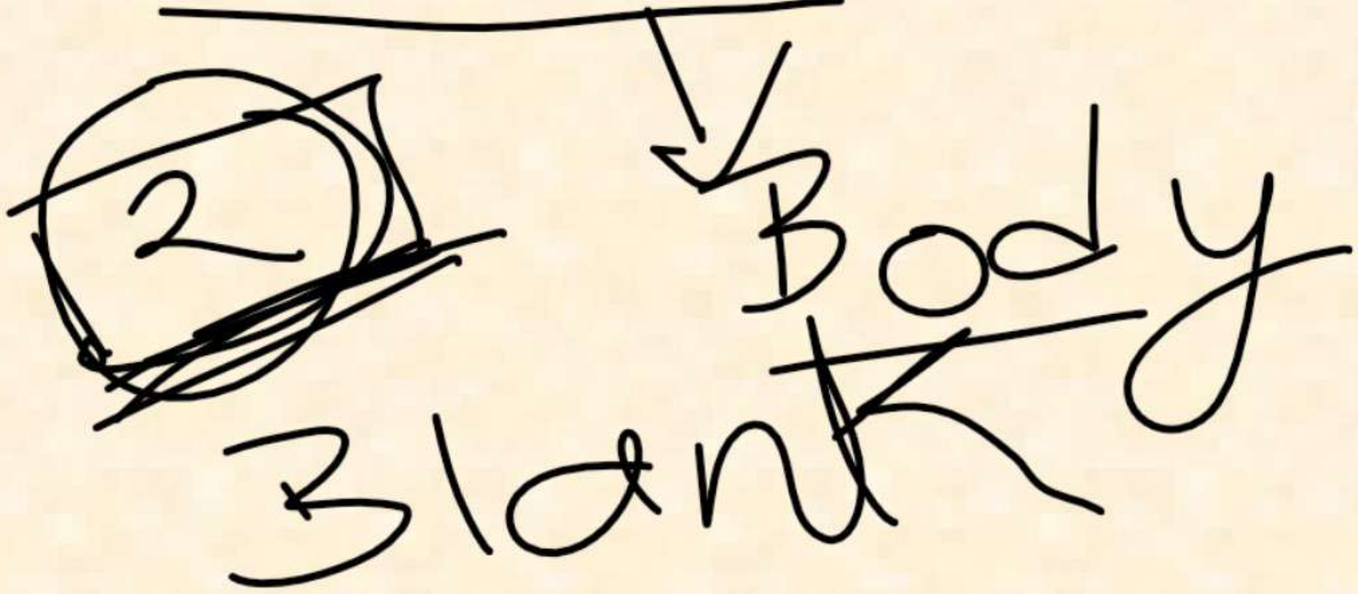


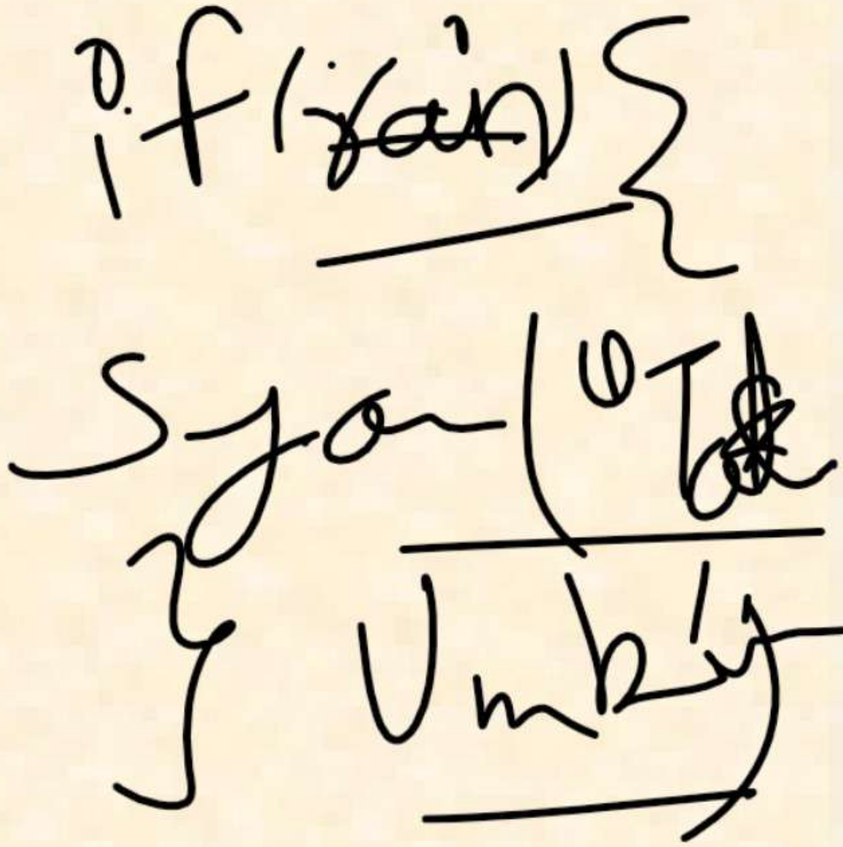
if Statement

Used when you want to execute a block of code **only if a condition is true**.

Syntax:

```
if (condition) {
        // code to be executed if condition is true
}
```



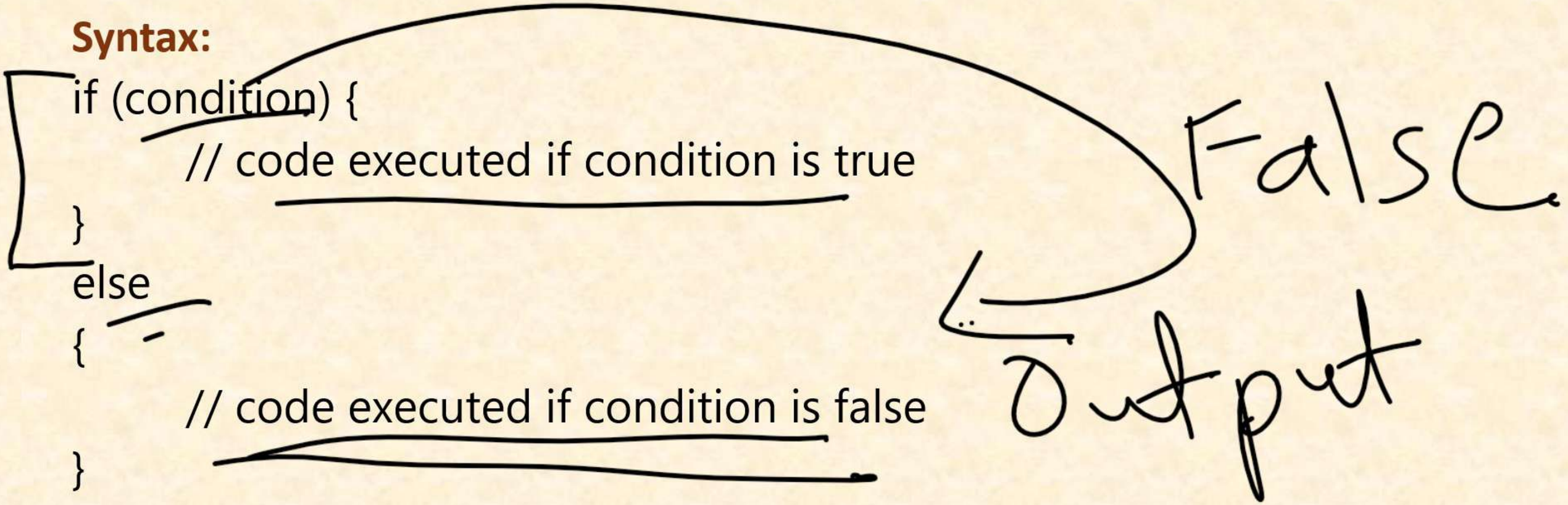


if-else Statement

Used when you want to choose between **two alternatives**.

Syntax:

```
if (condition) {
        // code executed if condition is true
}
else
{
        // code executed if condition is false
}
```

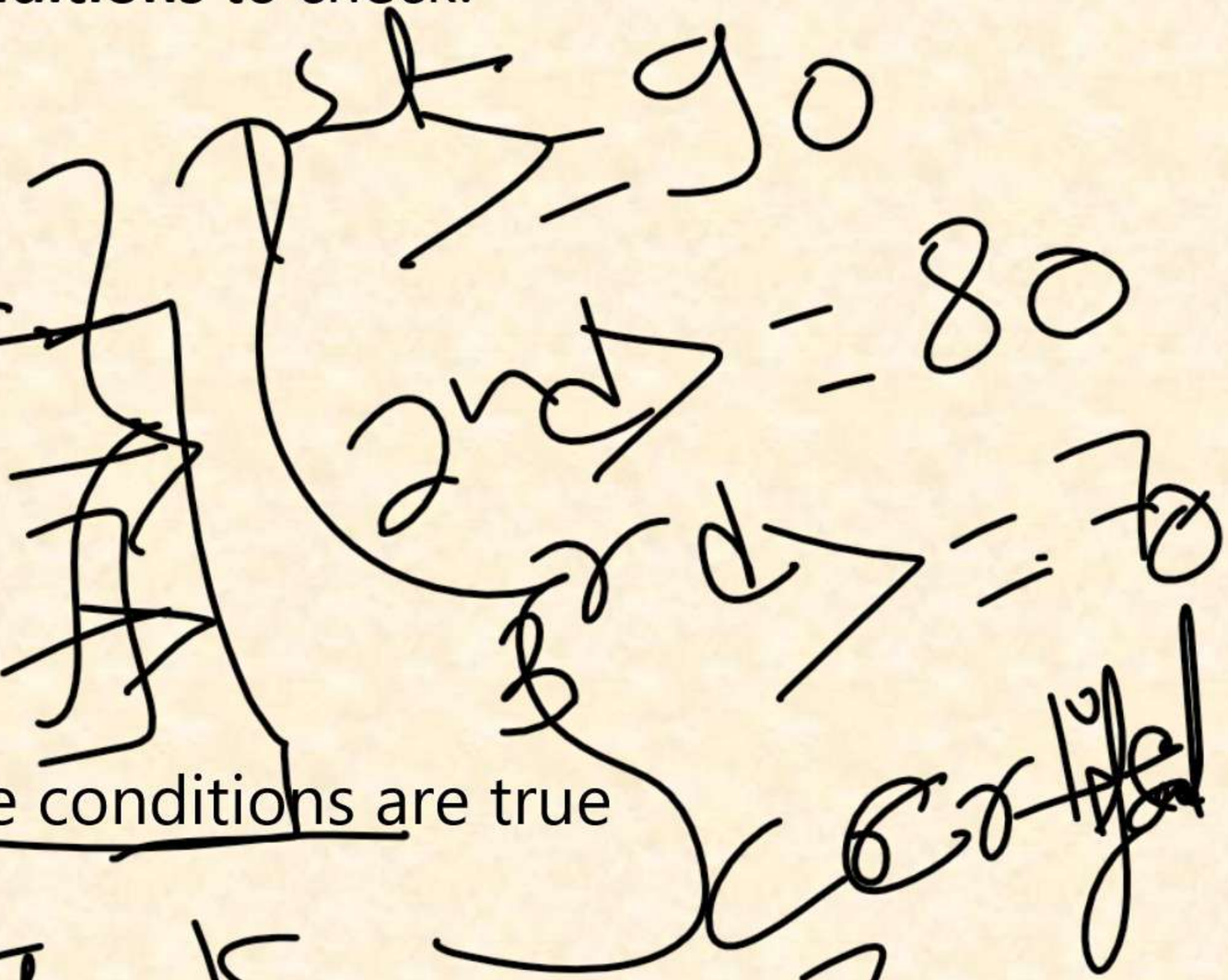


if-else-if Ladder

Used when there are **multiple conditions** to check.

Syntax:

```
if (condition1) {
    // code if condition1 is true
} else if (condition2) {
    // code if condition2 is true
} else if (condition3) {
    // code if condition3 is true
} else {
    // code if none of the above conditions are true
}
```

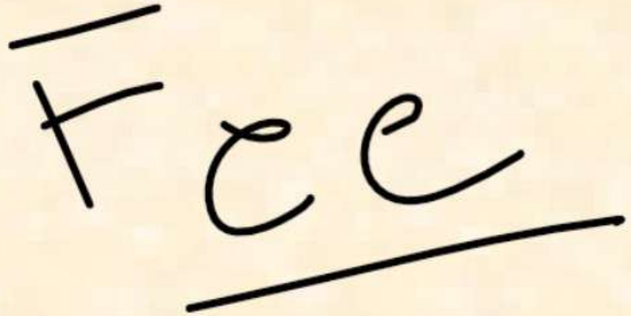


Nested if Statement

An if statement inside another if statement.

Syntax:

```
if (condition1) {
    if (condition2) {
        // code executed if both conditions are true
    }
}
```

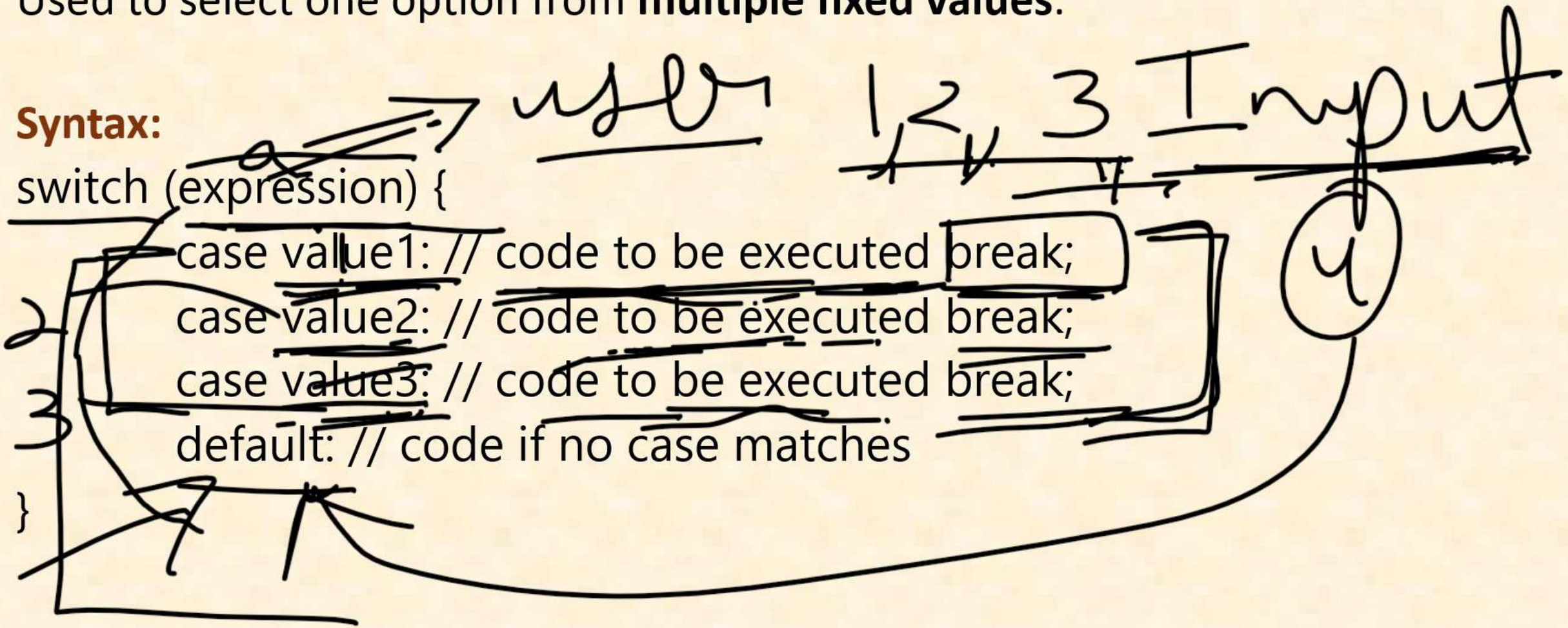


switch Statement

Used to select one option from **multiple fixed values**.

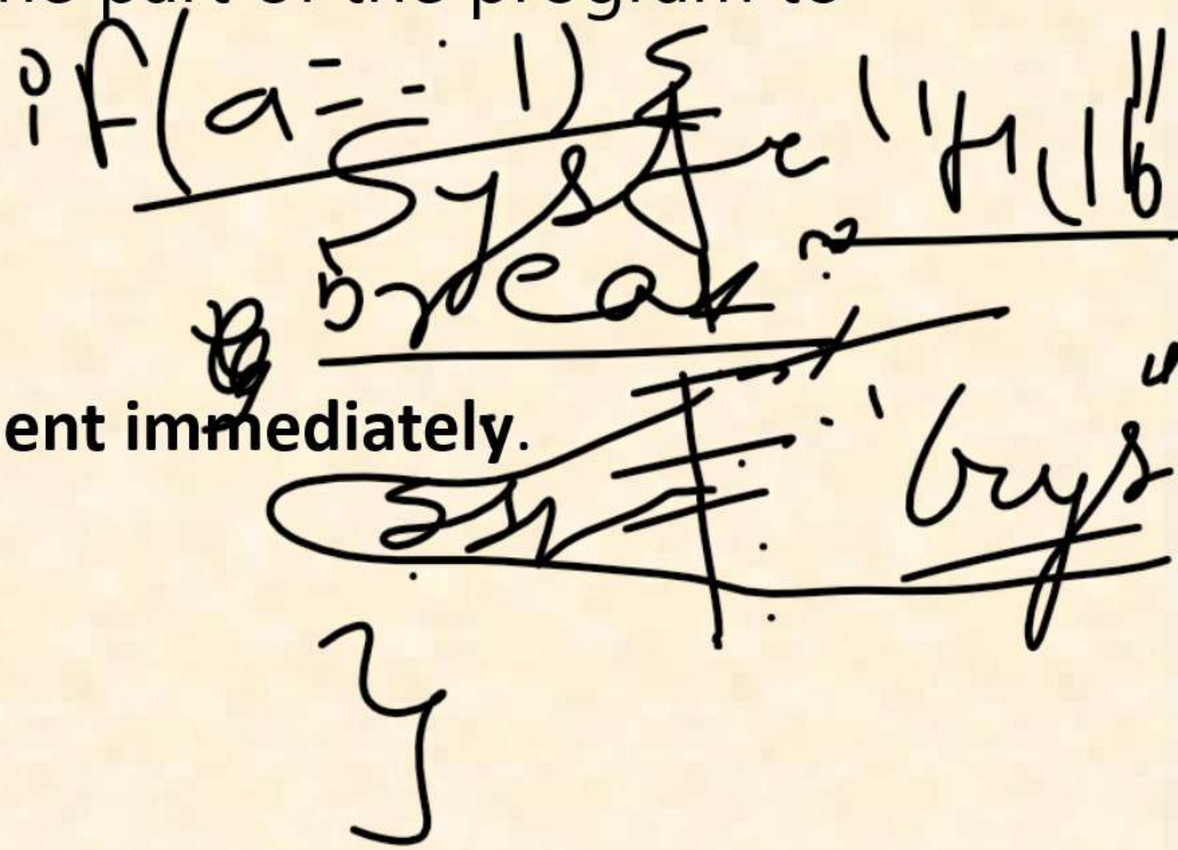
Syntax:

```
switch (expression) {
    case value1: // code to be executed break;
    case value2: // code to be executed break;
    case value3: // code to be executed break;
    default: // code if no case matches
}
```



Transfer Statements

Transfer statements are used to **change the normal flow of program execution**. They transfer control from one part of the program to another.



1.break Statement

Used to **terminate a loop or switch statement immediately**.

Syntax:

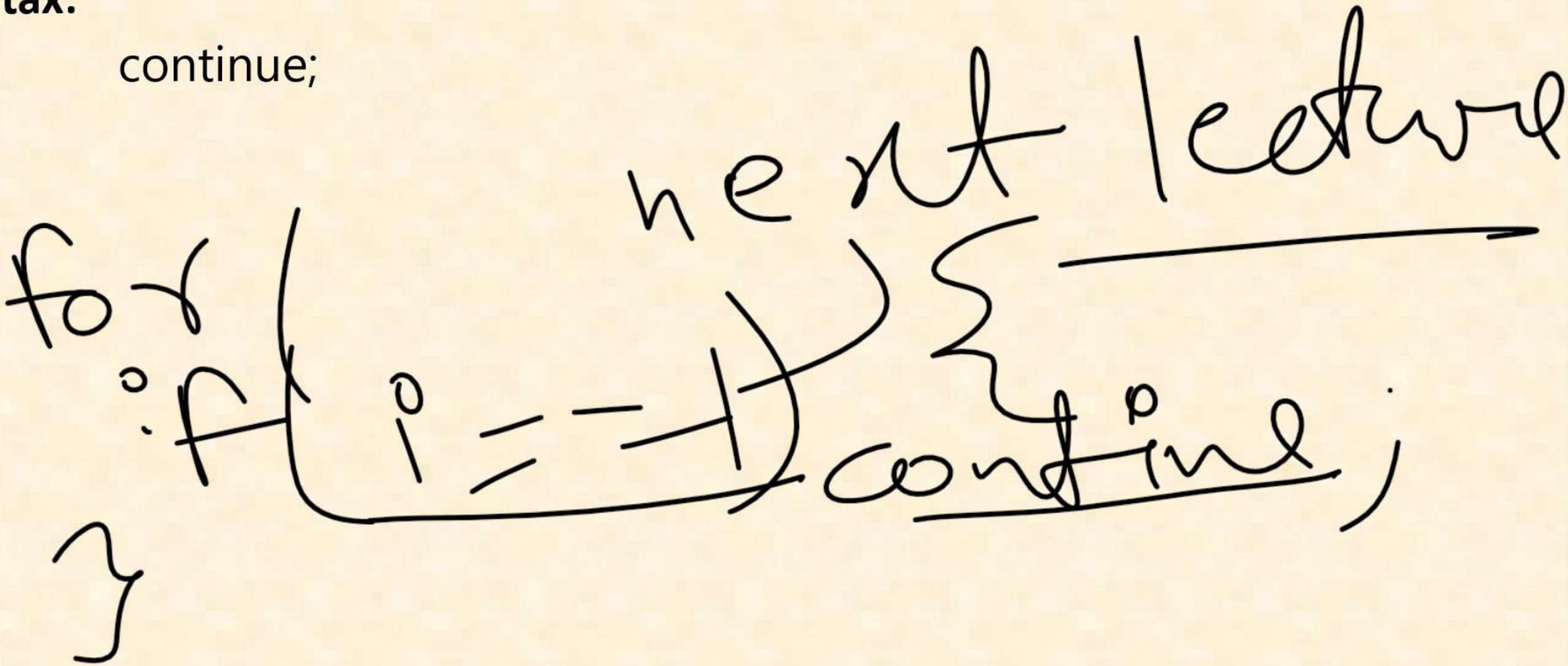
```
break;
```

2. continue Statement

Used to **skip the current iteration** of a loop and move to the next iteration.

Syntax:

```
continue;
```



3. return Statement

Used to **exit from a method** and optionally return a value to the calling method.

Syntax:

```
return;
return value;
return 0;
```