Reference card

Programmatic thinking

Comparison operators

Used to **compare numbers or strings** to perform the evaluation within a boolean expression.

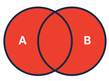
Equal to	==	Returns True when the value on the left is strictly equal to the value on the right.
Not equal to	!=	Returns True when the value on the left is strictly not equal to the value on the right.
Greater than	>	Returns True when the value on the left is greater than the value on the right.
Greater than or equal to	>=	Returns True when the value on the left is either greater than or equal to the value on the right.
Less than	<	Returns True when the value on the left is smaller than the value on the right.
Less than or equal to	<=	Returns True when the value on the left is either smaller than or equal to the value on the right.

Boolean operators

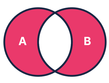
Used as conjunctions to **combine** (or **exclude**) **statements** in a boolean expression.

A NOT B — Only considers the argument after the operator. The algorithm considers the opposite of the argument.

A AND B — The two arguments are considered in conjunction. Both arguments should be True for the statement to be True.



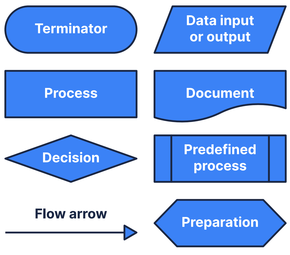
A OR B — If either of the arguments to the left or right of the operator is True, the statement will be True. If both arguments are True, the statement will also be True.



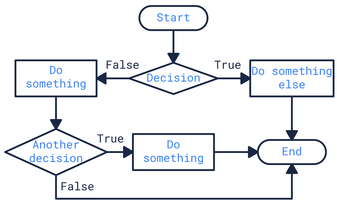
A XOR B — Only if exactly one of the arguments to the left or right of the operator is True will the statement be True. If both arguments are True, the statement is False.

Flowcharts

We use flowcharts (aka flow diagrams) to visually **represent the flow of control** of our logic, algorithms, pseudocode, and conditional statements.



Example of a flowchart:



Pseudocode

We can describe a **sequence of steps and actions** in a plain natural language called pseudocode. These are step-by-step descriptions for an algorithm using short but descriptive phrases.

```
if x % 2 == 0 then
   y = 1
else
   y = 0
```

Conditional statement

Represents decision-making by setting specific conditions.

```
if x % 2 == 0 then
   y = 1
else
   y = 0
```

Comparison operator

Compares numbers or strings to perform the evaluation within a boolean expression.

```
if x % 2 == 0 then
   y = 1
else
   y = 0
```

Boolean expression

A statement that results in an answer of either True or False.

```
if x % 2 == 0 then
   y = 1
else
   y = 0
```

Assignment operator

Assigns the value on the right to the variable on the left of the operator.

```
if x % 2 == 0 then
    y = 1
else
    y = 0
```

Indent

Groups statements that are intended to be executed together.

Conditional statements

If statement

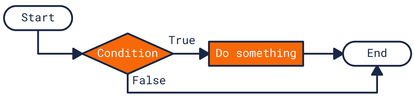
Pseudocode

```
START

If (condition) then
- Do something
End if

END
```

Flowchart



If-else statement

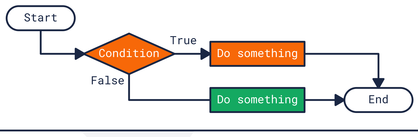
Pseudocode

```
START

If (condition) then
- Do something
Else
- Do something different
End if

END
```

Flowchart



Nested if statement

Pseudocode

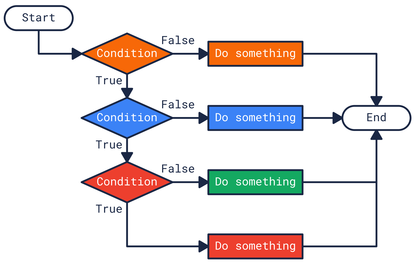
```
START

If (condition) then
- If (another condition) then
- - If (another condition) then
- - - Do something
- - Else
- - - Do something different
- Else
- - Do something different
Else
- Do something different
End if

END
```

Note: The succeeding conditions are only considered when the previous conditions are **True**.

Flowchart



If-else-if ladder

Pseudocode

```
START

If (condition) then
- Do something
Else if (another condition) then
- Do something different
Else if (another condition) then
- Do something different
Else
- Do something different
End if

END
```

Note: The succeeding conditions are only considered when the previous conditions are **False**.

Flowchart