### 'for' Loops

Learn about Kotlin's 'for' loops to iterate over any iterable structure, and how to build ranges for iteration.

# we'll cover the following ^ for Loops for Loops vs while Loops Quiz Exercises Summary

In contrast to while loops, Kotlin's for loops work a bit differently than you may be used to from other languages.

# for Loops #

Kotlin's for loops are different from many other languages because they're used in the same manner as *for-each loops* (or *for-in loops*). Thus, you always give an *iterable* to the for loop over which it iterates:

```
for (number in 1..5) println(number) // 1, 2, 3, 4, 5

for (number in 1 until 5) println(number) // 1, 2, 3, 4

for (number in 1..5 step 2) println(number) // 1, 3, 5

for (number in 5 downTo 1) println(number) // 5, 4, 3, 2, 1

for (number in 5 downTo 1 step 2) println(number) // 5, 3, 1

for (char in 'a'..'c') println(char) // 'a', 'b', 'c'

for (planet in planets) println(planet) // "Jupiter", "Saturn", ...

for (char in "Venus") println(char) // 'V', 'e', 'n', 'u', 's'
```





A few things to note here:

- *Ranges* are commonly used for basic loops, e.g., 1..5.
  - There are utilities to construct more complex ranges, e.g., downTo and step.
- Other common iterables are collections, e.g., planets.
- Kotlin's strings are also iterable.

**Note:** You can iterate over any object that implements **Iterable** using a for loop.

# for Loops vs while Loops #

As with if and when, you have multiple language constructs for one concept: repeating blocks of code. So when would you prefer for over while and vice versa?

- while loops are used when the *number of iterations is not known in advance*.
  - Consider again the approximation algorithm above. It may take any number of iterations to get below the required error margin because it's not a fixed number.
- for loops are superior when the *number of iterations* is known in advance.
  - In other words, the number of iterations is fixed because it's equal to the length of the given iterable.

## Quiz#

Kotlin's for loops



Which of the following can you iterate over using a for loop? You can select multiple answers.



### Exercises #

Implement a program that produces the following output *using loops*:

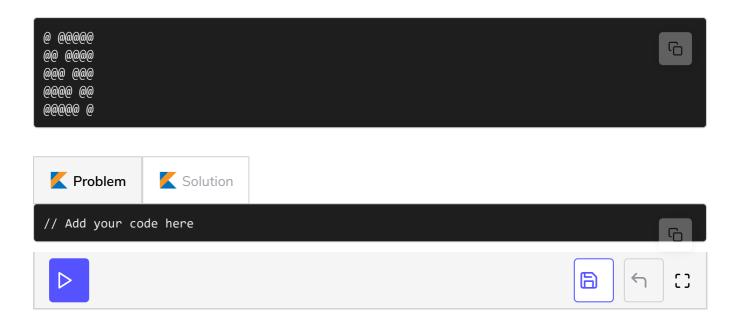


### **Hints:**

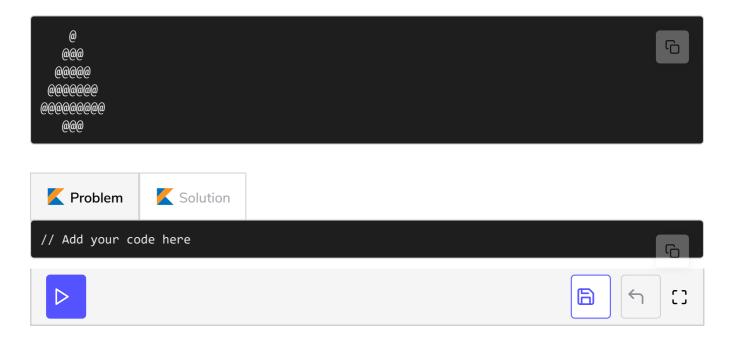
- You can nest loops, meaning that you can write a for loop inside another for loop.
- You can use print instead of println to avoid going to a new line after printing



Next, extend your program to produce the following output:



Finally, write a program that prints a Christmas tree:



### Summary #

Kotlin offers for, while, and do-while loops, each of which fits the bill for different use cases:

- Use for loops for a fixed number of iterations.
  - Kotlin offers many utilities that work well with for loops, such as ranges (1..10), downTo and step.
- Use while for a non-fixed number of iterations.
  - Use do-while if the first iteration initializes some data used in the

loop body.

Congrats, you now know how to use loops in Kotlin! In the next section, you will learn lots about functions in Kotlin, including using default values, infix functions, extension functions, and more.