1. Hello World.

For this introductory, non-Android Kotlin session, we are going to be using the online "Repl" site (`https://repl.it`) This allows you to write and run simple Kotlin programs in an IDE within the browser, and save them for later retrieval.
You can login to this using your GitHub, Google or Facebook account.
Logging in will allow you to save your work online, and retrieve it later.
Create a Hello World program. Write the code, and make sure you save it! Then, run it.

2. Loops and ifs – basic

a) Within repl, create a new program, `ex2.kt`. The program should ask the user to input their name and then should print it 10 times.
b) Modify the program so that it also asks how many times they want to display their name. HINT: `readLine()` might return null. See the section “A practical example, and introducing Elvis”

3. Looping through an array

a) Again within Repl, create a new program which stores an array of your favourite music artists.
The program should then prompt the user to ask the user to input an artist name. Use a loop to keep prompting the user until they guess one of the correct artists.

4. when

Write a program which uses a "`when`" statement to print the grade (A, B, C, D, E, or F) equivalent to a given percentage. Make the '`when`' act as an expression, i.e. you should get it to return a value and then print that value.
Grades are as follows:

A - 70-100
B - 60-69
C - 50-59
D - 40-49
E - 30-39
F – 0-29

The program should also display "Error - invalid percentage" if the percentage is below 0 or greater than 100.

Hint: you can use the "`in`" keyword with a range (e.g. 1..10) as a condition inside "`when`".

5. Classes and objects (do in your own time if necessary)

Create a `Car` class. It should have properties for make, model, engine capacity (cc), top speed, and current speed (mph or km/h - up to you). The first four should be passed in via the constructor.
Set the current speed to 0 when the car initialises.
Add `accelerate()` and `decelerate()` methods. These should increase and decrease the speed by a value passed into the method as a parameter. decelerate() should ideally check to make sure that the speed doesn't go below 0 and `accelerate()` should ideally check to make sure that the speed doesn't go above the top speed.
Add a `toString()` method to return a string representation of the car. You need to use the `override` keyword with this, because it’s overriding the superclass method, i.e.

```
override fun toString(): String {...
}
```

Write a `main()` function. In `main()`, try creating two car objects. Display the car objects, accelerate them, display them again, decelerate them again, and display them once more

6. Inheritance (advanced; do in your own time if necessary)

You need to look at the further notes to answer this question.
Create an inheritance hierarchy involving employees, programmers and managers.
The `Employee` class should have attributes for name, job title and salary.
The `Programmer` class, inheriting from `Employee`, should in addition have an attribute for favourite language (which should be a `String`) and current project (a `String`).
The `Manager` class, also inheriting from `Employee`, should in addition attributes for `companyCar` (a `Car` - borrow the class from the previous exercise) and number of shares in the company.
Give each class a `printDetails()` method which prints the full details of objects of that class.
Write a `main()` function to test your inheritance hierarchy.
Enhancement: change `Programmer` so that the favourite language string is replaced by an array of strings for multiple favourite languages (e.g. Kotlin, Java, PHP, Python and JavaScript).