COMP8710 Class 4.2 – Behavioural parameterisation

Introduction

You may have noticed that some of the functions you implemented in Class 4.1 had a similar structure. We are going to use behavioural parametrisation to make the code cleaner.

Setup

Open your **MusicTracks** project from last week again. You may want to take a copy for safe-keeping first; we will modify the project now, and maybe you would like to keep a copy of last week's solution.

Task

Taking inspiration from the "Filtering Apples" example we saw in the lectures, implement new *higher-order* methods[1] and, if required, new *functional interfaces*[2] which you can invoke in cleaner implementations of printList, printTracksWith, etc. Feel free to create new classes and interfaces as required.

Hint: *Try to move most of the for loops outside of the public methods you implemented in the previous class.*

Finishing

Reflect on the task you have achieved today. What was particularly challenging? Are the newly defined methods easy to use?

[1] Functions that take other functions as parameter.

[2]Don't forget the @FunctionalInterface tag!