正如本小节开头所说的，这里我们重点学习的是函数式 API 的语法结构，理解了语法结构之后，集合中的各种其他函数式 API 都是可以快速掌握的。

接下来我们就再来学习几个集合中比较常用的函数式 API，相信这些对于现在的你来说，应该是没有什么困难的。

集合中的 map 函数是最常用的一种函数式 API，它用于将集合中的每个元素都映射成一个另外的值，映射的规则在 Lambda 表达式中指定，最终生成一个新的集合。比如，这里我们希望让所有的水果名都变成大写模式，就可以这样写：

```
fun main() {
    val list = listOf("Apple", "Banana", "Orange", "Pear", "Grape", "Watermelon")
    val newList = list.map { it.toUpperCase() }
    for (fruit in newList) {
        println(fruit)
    }
}
```

可以看到，我们在 map 函数的 Lambda 表达式中指定将单词转换成了大写模式，然后遍历这个新生成的集合。运行一下代码，结果如图 2.26 所示。

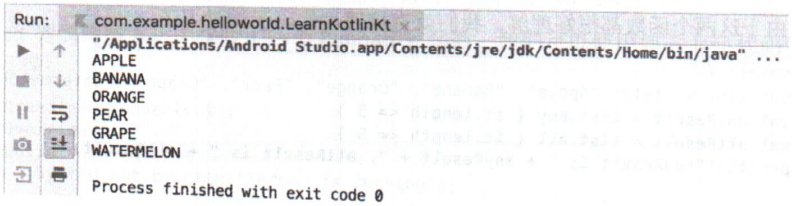

图 2.26 将水果名都转换成大写模式

map 函数的功能非常强大，它可以按照我们的需求对集合中的元素进行任意的映射转换，上面只是一个简单的示例而已。除此之外，你还可以将水果名全部转换成小写，或者是只取单词的首字母，甚至是转换成单词长度这样一个数字集合，只要在 Lambda 表示式中编写你需要的逻辑即可。

接下来我们再来学习另外一个比较常用的函数式 API——filter 函数。顾名思义，filter 函数是用来过滤集合中的数据的，它可以单独使用，也可以配合刚才的 map 函数一起使用。

比如我们只想保留 5 个字母以内的水果，就可以借助 filter 函数来实现，代码如下所示：

```
fun main() {
    val list = listOf("Apple", "Banana", "Orange", "Pear", "Grape", "Watermelon")
    val newList = list.filter { it.length <= 5 }
                      .map { it.toUpperCase() }
    for (fruit in newList) {
        println(fruit)
    }
}
```