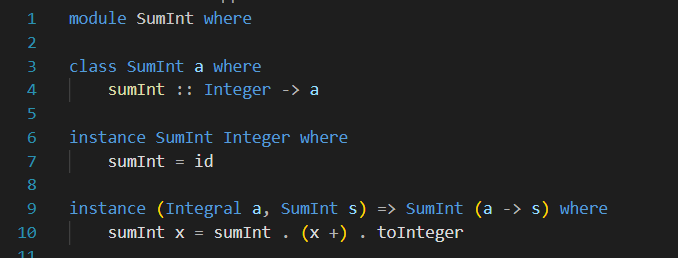
# 问题：实现实现多参数求和sum函数

## 实现多Int参数的求和函数sumInt

该函数可以接受多个Int类型的参数，返回它们的和。

可以使用类型机制来实现。实现代码如下：



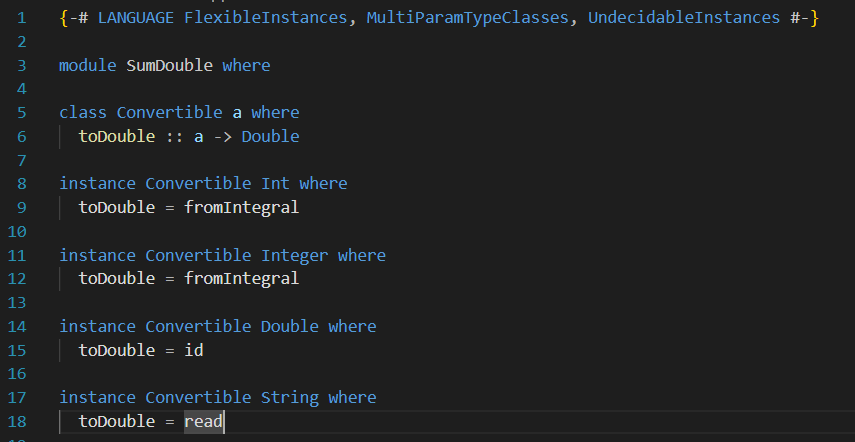
* Class SumInt ：定义类型
* Integer的实例化：在递归最后只有一个整数时，直接返回。
* (Integer -> SumInt)的实例化：函数f接受一个整数参数a，返回一个SumInt类型（或整数，即a+x；或（Integer->SumInt）的函数，即一个+x的函数）

通过以上迭代，就可以计算多参数求和。在调用时，要指定函数返回值的类型为Int。不然编译器可能会推断该返回值是一个（Integer->SumInt）函数类型，从而报错。

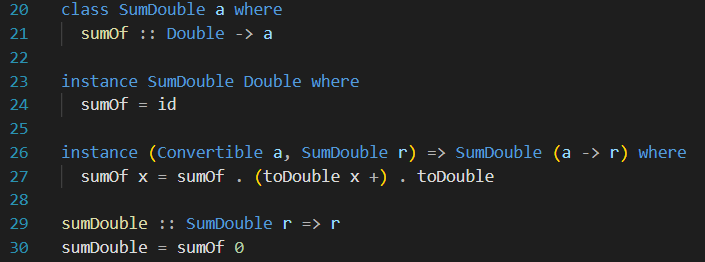
## 实现多类型参数，返回double类型的求和函数sumDouble

该函数可以接受任意多个支持向double转化的类型作为参数，最后返回所有转化后的浮点数之和。

首先需要定义可转化为double的类，并提供转化方法。目前支持Int/Integer/Double/String：



然后只需要再定义SumDouble类即可，与SumInt类似，只需要再添加(Convertible -> SumInt)的实例化。

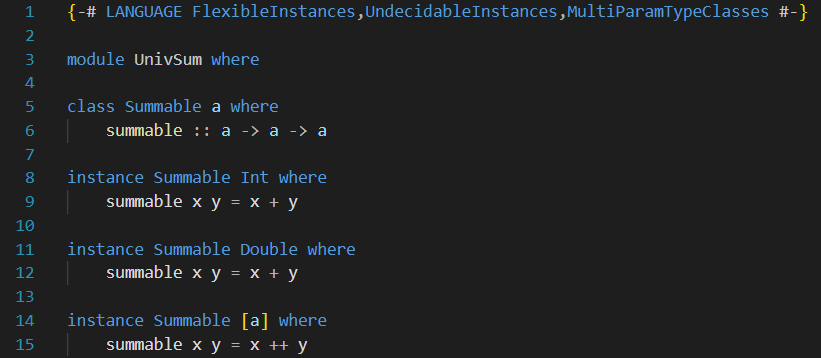


最后需要一个辅助函数sumDouble而不能直接使用类的方法sumOf，这是因为sumOf显式地指定了输入参数应该是Double类型。如果直接使用sumOf，那么在使用时的第一个参数就必须是Double类型而不能是其他类型。所以添加一个辅助函数sumDouble，在其中调用sumOf 0，于是能够去除sumOf的类型限制。

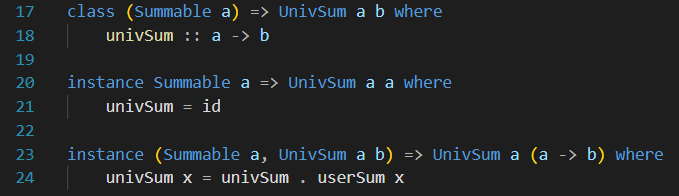
## 实现可求和类型的多参数求和函数univSum

该函数可以接受任意数量的可求和类型参数。对于Int/Double等数值类型，返回求和；对String/Array等类型，返回拼接的结果。

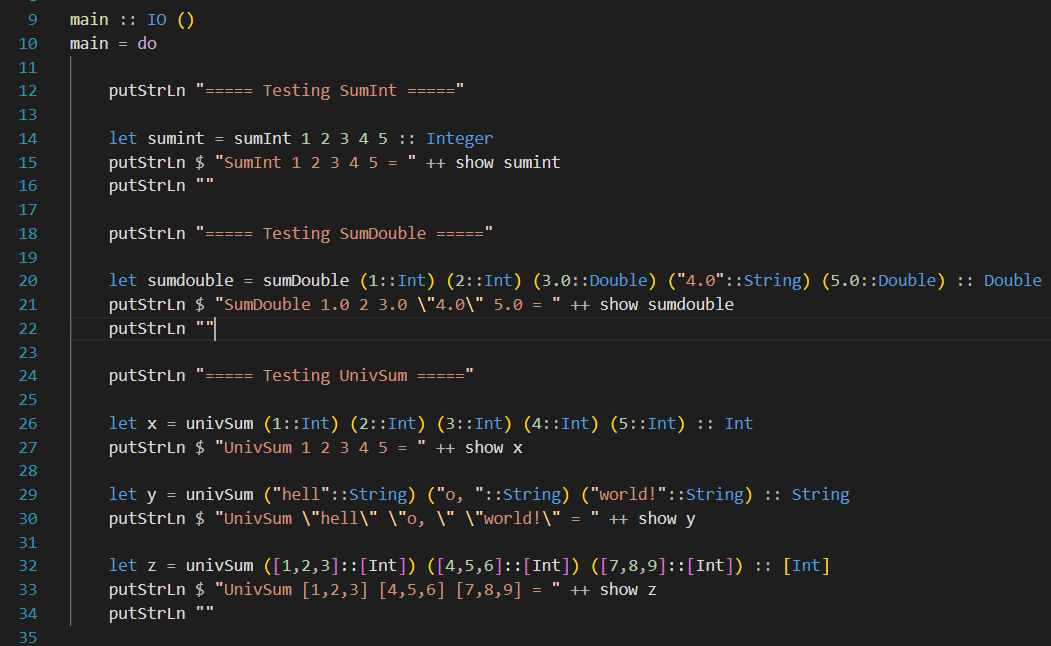
由于需要对任意类型求和，所以需要自己定义一个可求和类Summable，并定义userSum的自定义求和方法。目前能够实现Int/Double/Array(String)的求和。定义如下：



接下来与SumInt/SumDouble类似，也是通过类型机制实现可变参数。但是由于类的辅助函数的定义需要依赖Summable，所以需要引入MultiParamTypeClasses的语言选项。代码如下：



## 测试代码



测试代码在main函数中。

## 测试结果

