高阶函数

- 定义
  - 将一个或者多个函数当作参数
  - 把一个函数当作返回值
  - 满足一个即可
- map
  - 通过某种规则（闭包实现）进行转换
    - 数组
      - map源码
        - 1，构造一个名为result且与原数组的 capacity一致的新数组，用于存放新的结果
        - 2，遍历自己的元素，对每个元素调用闭包的转换函数，进行转换
        - 3，将转换的结果使用 append方法存入result中
        - 4，遍历完成后，返回result。
        - 5,返回结果是 [T]
    - Option
  - Sequence协议
    - Element
      - 序列里面的元素
    - Iterator
      - 迭代器
      - 通过next() 获取元素
  - map和flatMap的transform方法的区别
    - append(transform(T))
      - 直接将元素放入result
    - append(contentOf:transform(element))
      - flatMap将数组中的元素一一取出来，更准确的说是调用Sequence的迭代器的next()方法，获取元素，然后放入result。
- flatMap
  - 通过某种规则(闭包实现)进行转换
    - 数组
      - 将多维数组降一维，4维数组转化为3维数组
        - flatMap源码
          - 1，创建一个新数组
          - 2，遍历自己的元素，对每个元素调用闭包的转换函数，进行转换
          - 3，将转换的结果使用 append(contentOf)，存入result中
          - 4，遍历完成后，返回result。
          - 5,返回结果为[Sequece.Element]
- compactMap
  - 在4.1之后对flatMap一个重载方法的重命名
    - 1，会剔除nil对象
    - 2，会对Optional 进行解包
    - 通过 if let 实现
  - 源码
- filter
  - 通过Bool表达式筛选
    - 源码
      - 1，调用_filter方法，传入 isIncluded函数
      - 2，构造一个新数组，用于存放新结果
      - 3，使用 迭代器，遍历所有的元素，对每个元素调用 isInclude函数，判断是否符合条件
      - 4，将符合条件的元素 使用 append 方法放入result 中
      - 5，遍历完成后，返回result
- reduce
  - 定义
    - 将数组元素通过一定规则，组合计算为另一个值
    - 接收一个初始值，初始值的类型可以和数组中元素类型不一致
  - 基础思想
    - 将一个序列转换为一个不同类型的数据，通过accumulator来记录递增状态
  - 示例
  - 源码
    - 1，定义 accumulator临时变量，并赋值initialResult。
    - 2,遍历所有的元素，对每个元素调用闭包updateAccumulatingResult，通过 inout 参数，更新 accumulator的值。
    - 3，遍历完成后，返回 accumulator
  - 拓展
    - 实现map
    - 实现filter
    - 实现flatMap
    - 实现compactMap