**Scala程序设计解析**

**配套课后习题**

**目 录**

[第1章 Scala基础 2](#_Toc492826211)

[第2章 控制结构和函数 2](#_Toc492826212)

[第3章 数据结构 3](#_Toc492826213)

[第4章 模式匹配 4](#_Toc492826214)

[第5章 高阶函数 6](#_Toc492826215)

[第6章 类 7](#_Toc492826216)

[第7章 对象 7](#_Toc492826217)

[第8章 包和引入 8](#_Toc492826218)

[第9章 继承 8](#_Toc492826219)

[第10章 特质 9](#_Toc492826220)

[第11章 类型参数 9](#_Toc492826221)

[第12章 文件和正则表达式 10](#_Toc492826222)

[第13章 高级类型 10](#_Toc492826223)

# Scala基础

**1、在Scala REPL中，计算3的平方根,然后再对该值求平方。现在，这个结果与3相差多少？**

**2、如何检测一个变量是val还是var?**

**3、Scala允许你用数字去乘一个字符串，去REPL中试一下"crazy"\*3。这个操作做什么？在Scaladoc中如何找到这个操作?**

**4、10 max 2的含义是什么？max方法定义在哪个类中？**  
  
**5、用BigInt计算2的1024次方**

**6、在Scala中如何获取字符串“Hello”的首字符和尾字符？**

# 控制结构和函数

**1、一个数字如果为正数，则它的signum为1;如果是负数,则signum为-1;如果为0,则signum为0.编写一个函数来计算这个值**

**2、一个空的块表达式{}的值是什么？类型是什么？**

**3、针对下列Java循环编写一个Scala版本:**

**for(int i=10;i>=0;i–)System.out.println(i);**

**4、编写一个过程countdown(n:Int)，打印从n到0的数字**

**5、编写一个for循环,计算字符串中所有字母的Unicode代码（toLong方法）的乘积。举例来说，"Hello"中所有字符串的乘积为9415087488L**

**6、同样是解决前一个练习的问题，请用StringOps的foreach方式解决。**

**7、编写一个函数product(s:String)，计算字符串中所有字母的Unicode代码（toLong方法）的乘积**

**8、把7练习中的函数改成递归函数**

**9、编写函数计算,其中n是整数，使用如下的递归定义:**

* ** = x\*,如果n是正数的话**
* ** = 1**
* ** = 1/,如果n是负数的话**
* **不得使用return语句**

# 数据结构

**1、编写一段代码，将a设置为一个n个随机整数的数组，要求随机数介于0和n之间。**

**2. 编写一个循环，将整数数组中相邻的元素置换。比如**Array(1, 2, 3, 4, 5)置换后为Array(2, 1, 4, 3, 5)

**3. 给定一个整数数组，产出一个新的数组，包含原数组中的所有正值，以原有顺序排列，之后的元素是所有零或负值，以原有顺序排列。**

**4. 创建一个由java.util.TimeZone.getAvailableIDs返回的时区集合，并只显示以America/前缀开头的时区，并且有序。**

**5、设置一个映射，其中包含你想要的一些装备，以及它们的价格。然后根据这个映射构建另一个新映射，采用同一组键，但是价格上打9折。**

**6、编写一段WordCount函数，统计传入的字符串中单词的个数**

**7、重复上一个练习，使统计后的单词有序**

**8、 重复前一个练习，使用java.util.TreeMap进行实现，并使之适用于Scala API**

**9、在REPL中打印出所有Java系统属性的表格，需要有格式，如下：**

****

**10、编写一个函数 minmax(values:Array[Int]), 返回数组中最小值和最大值的对偶**

**11、编写一个函数indexes，给定字符串，产出一个包含所有字符下标的映射。举例来说：indexes(“Mississippi”)应返回一个映射，让’M’对应集{0}，‘i’对应集{1，4，7，10}，依次类推。使用字符到可变集的映射，注意下标的集应该是有序的。**

**12、编写一个函数，从一个整型链表中去除所有的零值。**

**13、编写一个函数，接受一个字符串的集合，以及一个从字符串到整数值的映射。返回整形的集合，其值为能和集合中某个字符串相对应的映射的值。举例来说，给定Array(“Tom”,”Fred”,”Harry”)和Map(“Tom”->3,”Dick”->4,”Harry”->5)，返回Array(3,5)。提示：用flatMap将get返回的Option值组合在一起。**

**14、实现一个函数，作用与mkStirng相同，提示：使用reduceLeft实现试试**

**15、给定整型列表lst，(lst :\ List[Int]())(\_ :: \_)得到什么? (List[Int]() /: lst)(\_ :+ \_)又得到什么？如何修改他们中的一个，以对原列表进行反向排列?**

**16、编写一个函数，将Double数组转换成二维数组。传入列数作为参数。距离来说，传入Array(1,2,3,4,5,6)和3列，返回Array(Array(1,2,3), Array(4,5,6))。**

**17【选做】\*通过par编写一个并行的wordcount程序，统计字符串中每个字符的出现频率。可以通过并行集合的aggregate函数试试**

# 模式匹配

**1、利用模式匹配，编写一个swap函数，接受一个整数的对偶，返回对偶的两个组成部件互换位置的新对偶**

**2、利用模式匹配，编写一个swap函数，交换数组中的前两个元素的位置，前提条件是数组长度至少为2**

**3、编写计算Item价格的通用函数。**

**abstract class** Item  
**case class** Article(description: String, price: Double) **extends** Item  
**case class** Bundle(description: String, discount: Double, items: Item\*) **extends** Item  
  
**val** special = *Bundle*(**"Father's day special"**, 20.0,  
 *Article*(**"Scala for the Impatient"**, 39.95),  
 *Bundle*(**"Anchor Distillery Sampler"**, 10.0,  
 *Article*(**"Old Potrero Straight Rye Whiskey"**, 79.95),  
 *Article*(**"Junípero Gin"**, 32.95)))

**4、通过case类制作一颗二叉树。**   
**sealed abstract class BinaryTree  
case class Leaf(value : Int) extends BinaryTree  
case class Node(left : BinaryTree,right : BinaryTree) extends BinaryTree**  
**编写一个通用函数计算所有叶子节点中的元素之和。**

**5、扩展前一个练习中的树，使得每个节点可以有任意多的后代，并重新实现计算函数。完成下面这棵树的叶子节点元素统计：Node(Node(Leaf(3),Leaf(8)),Leaf(2),Node(Leaf(5)))**

**6、扩展前一个练习中的树，使得每个非叶子节点除了后代之外，能够存放一个操作符。然后编写一个eval函数来计算它的值。举例来说：**

**上面这棵树的值为(3 \* 8) + 2 + (-5) = 21**  
**新的case类如下：**

**sealed abstract class BinaryTree**

**case class Leaf(value: Int) extends BinaryTree**

**case class Node(op: Char, leafs: BinaryTree\*) extends BinaryTree**

**完成下面的树的计算：**

**Node('+', Node('\*', Leaf(3), Leaf(8)), Leaf(2), Node('-', Leaf(5)))**

**7、编写一个函数，计算List[Option[Int]]中所有非None值之和。不得使用match语句。**

# 高阶函数

**1、编写一个compose函数，将两个类型为Double=>Option[Double]的函数组合在一起，产生另一个同样类型的函数。如果其中一个函数返回None，则组合函数也应返回None。例如：**  
**def f(x : Double) = if ( x >= 0) Some(sqrt(x)) else None  
def g(x : Double) = if ( x != 1) Some( 1 / ( x - 1)) else None  
val h = compose(f,g)  
h(2)将得到Some(1)，而h(1)和h(0)将得到None**

**2、编写函数values(fun:(Int)=>Int,low:Int,high:Int),该函数输出一个集合，对应给定区间内给定函数的输入和输出。比如，values(x=>x\*x,-5,5)应该产出一个对偶的集合(-5,25),(-4,16),(-3,9),…,(5,25)**

**3、如何用reduceLeft得到数组Array(1,333,4,6,4,4,9,32,6,9,0,2)中的最大元素?**

**4、用to和reduceLeft实现阶乘函数,不得使用循环或递归**

**5、编写函数largest(fun:(Int)=>Int,inputs:Seq[Int]),输出在给定输入序列中给定函数的最大值。举例来说，largest(x=>10x-xx,1 to 10)应该返回25.不得使用循环或递归**

**6【选做】、修改前一个函数，返回最大的输出对应的输入。举例来说,largestAt(fun:(Int)=>Int,inputs:Seq[Int])应该返回5。不得使用循环或递归**

**7、要得到一个序列的对偶很容易，比如:**  
**val pairs = (1 to 10) zip (11 to 20)**

**编写函数adjustToPair,该函数接受一个类型为(Int,Int)=>Int的函数作为参数，并返回一个等效的, 可以以对偶作为参数的函数。举例来说就是:adjustToPair(\_\*\_)((6,7))应得到42。然后用这个函数通过map计算出各个对偶的元素之和**

**8、实现一个unless控制抽象，工作机制类似if,但条件是反过来的。**

# 类

**1、编写一个Time类，加入只读属性hours和minutes，和一个检查某一时刻是否早于另一时刻的方法before(other:Time):Boolean。Time对象应该以new Time(hrs,min)方式构建。**

**2、创建一个Student类，加入可读写的JavaBeans属性name(类型为String)和id(类型为Long)。有哪些方法被生产？(用javap查看。)你可以在Scala中调用JavaBeans的getter和setter方法吗？**

# 对象

**1、编写一个Conversions对象，加入 inchesToCentimeters,gallonsToLiters 和 milesToKilometers 方法**

**2、定义一个 Point 类和一个伴生对象,使得我们可以不用 new 而直接用 Point(3,4)来构造 Point 实例 apply 方法的使用**

**3、编写一个 Scala 应用程序,使用 App 特质,以反序打印命令行参数,用空格隔开。举例来说,scala Reverse Hello World 应该打印 World Hello**

**4、编写一个扑克牌 4 种花色的枚举,让其 toString 方法分别返回♣,♦,♥,♠，并实现一个函数,检查某张牌的花色是否为红色**

# 包和引入

**1、练习使用包的各种声明方式，并查看他们的不同**

**2、编写一段程序,将Java哈希映射中的所有元素拷贝到Scala哈希映射。用引入语句重命名这两个类。**

# 继承

**1、扩展如下的BankAccount类，新类CheckingAccount对每次存款和取款都收取1美元的手续费**  
**class BankAccount(initialBalance:Double){  
 private var balance = initialBalance  
 def deposit(amount:Double) = { balance += amount; balance}  
 def withdraw(amount:Double) = {balance -= amount; balance}  
}**

**2、扩展前一个练习的BankAccount类，新类SavingsAccount每个月都有利息产生(earnMonthlyInterest方法被调用)，并且有每月三次免手续费的存款或取款。在earnMonthlyInterest方法中重置交易计数。**

**3、定义一个抽象类Item,加入方法price和description。SimpleItem是一个在构造器中给出价格和描述的物件。利用val可以重写def这个事实。Bundle是一个可以包含其他物件的物件。其价格是打包中所有物件的价格之和。同时提供一个将物件添加到打包当中的机制，以及一个适合的description方法**

**4、设计一个Point类，其x和y坐标可以通过构造器提供。提供一个子类LabeledPoint，其构造器接受一个标签值和x,y坐标,比如:new LabeledPoint(“Black Thursday”,1929,230.07)**

**5、定义一个抽象类Shape，一个抽象方法centerPoint，以及该抽象类的子类Rectangle和Circle。为子类提供合适的构造器，并重写centerPoint方法**

**6、提供一个Square类，扩展自java.awt.Rectangle并且是三个构造器：一个以给定的端点和宽度构造正方形，一个以(0,0)为端点和给定的宽度构造正方形，一个以(0,0)为端点,0为宽度构造正方形**

# 特质

**1、java.awt.Rectangle类有两个很有用的方法translate和grow,但可惜的是像java.awt.geom.Ellipse2D这样的类没有。在Scala中，你可以解决掉这个问题。定义一个RenctangleLike特质,加入具体的translate和grow方法。提供任何你需要用来实现的抽象方法,以便你可以像如下代码这样混入该特质:  
val egg = new java.awt.geom.Ellipse2D.Double(5,10,20,30) with RectangleLike  
egg.translate(10,-10)  
egg.grow(10,20)**

# 类型参数

**1、定义一个不可变类Pair[T,S], 带一个swap方法，返回组件交换过位置的新对偶**

2、**定义一个可变类Pair[T]，带一个swap方法，交换对偶中组件的位置。**

**3、给定类Pair[T, S] ，编写一个泛型方法swap，接受对偶作为参数并返回组件交换过位置的新对偶。**

**4、编写一个泛型方法middle，返回任何Iterable[T]的中间元素。举例来说，middle(“World”)应得到’r’。**

**5、给定可变类Pair[S,T]，使用类型约束定义一个swap方法，当类型参数相同时可以被调用。**

# 文件和正则表达式

**1、编写一小段Scala代码，将某个文件中的行倒转顺序(将最后一行作为第一行,依此类推)**

**2、编写Scala程序打印出某个网页中所有img标签的src属性。使用正则表达式和分组**

# 高级类型

**1、实现一个Bug类，对沿着水平线爬行的虫子建模。move方法向当前方向移动，turn方法让虫子转身，show方法打印出当前的位置。让这些方法可以被串接调用。例如：  
bugsy.move(4).show().move(6).show().turn().move(5).show()  
上述代码应显示4 10 5。**

2、**为前一个练习中的Bug类提供一个流利接口，达到能编写如下代码的效果：  
bugsy move 4 and show and then move 6 and show turn around move 5 and show**

**3、实现一个方法，接受任何具备如下方法的类的对象和一个处理该对象的函数。  
调用该函数，并在完成或有任何异常发生时调用close方法。**  
**def close(): Unit**

4、**编写一个函数printValues，带有三个参数f、from和to，打印出所有给定区间范围内的输入值经过f计算后的结果。这里的f应该是任何带有接受Int产出Int的apply方法的对象。例如：  
printValues((x: Int) => x\*x, 3, 6) //将打印 9 16 25 36  
printValues(Array(1, 1, 2, 3, 5, 8, 13, 21, 34, 55), 3, 6) //将打印 3 5 8 13**