

面向对象编程：Java中的抽象数据类型 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/145/2021_2022__E9_9D_A2_E5_90_91_E5_AF_B9_E8_c104_145421.htm 在本文中，我们将考察Java中的数据类型，但是我们将介绍抽象数据类型（ADT）的概念。我们还将通过介绍Java Collections Framework（Java集合架构）来学习Java定义的一些ADT。ADT 一个ADT是一个仅由保存的数据类型和可能在这个数据类型上进行的操作定义的。开发者们只能通过ADT的操作方法来访问ADT的属性，而且他们不会知道这个数据类型内部各种操作是如何实现的。在Java中，我们常常使用一个接口来给出一个操作集合而不需要透露这些操作实现的细节。记住一个接口定义了一个方法集而Java类必须实现这个集合以便满足它的强制性条件或者实现这个接口的一个实例。线性表，堆栈和队列当我们谈论ADT的时候，经常会说到线性表，堆栈和队列。我们不会讨论这些数据结构的细节，但我们会讨论为什么它们被称为ADT。一个线性表是有限个元素的集合，其元素以线性的方式进行排列并提供对它的元素的直接访问。一个堆栈是一个后进先出（LIFO）的有序线性表，元素从堆栈头加入，并从堆栈头取出。一个队列是一个先进先出的有序线性表，元素从队列尾加入，并从队列头取出。线性表，堆栈和队列的内部结构可以用许多方式实现。例如，我们可以使用一个有序数组或者一个链表来实现每个结构。关键的一点是不论你如何实现其内部结构，它对外的接口总是不变的。这使得你能够修改或者升级底层的实现过程而不需要改变公共接口部分。Java集合架构 Java 2软件开发包（SDK）提供了一些新

类来支持大多数常用的ADT。这些类被称为Java集合类（类似于MFC中的集合类），它们协同工作从而形成Java集合架构。这个集合架构提供了一套将数据表示成所谓的集合抽象数据的接口和类。java.util.Collection接口被用来表示任意的成组的对象，也就是元素。这个接口提供基本的诸如添加，删除，和查询这样的操作。Collection接口还提供了一个iterator方法。iterator方法返回java.util.Iterator接口的一个实例。

而Iterator接口又提供了hasNext, next, 和 remove方法。使用Iterator接口提供的方法，你可以从头到尾循环遍历一个Collection对象中的实例并能够安全的删除iterator(游标)所表示的元素。java.util.AbstractCollection是所有集合架构类的基础。AbstractCollection类提供了对java.util.Collection接口中除iterator和size方法以外的所有方法的实现。这两个例外的方法由所有继承java.util.AbstractCollection的子类实现。实现一个接口的类必须提供对所有接口方法的实现。因为集合架构中的一些接口方法是可选的，所以必须有一种方法来通知调用者某种方法没有实现。当一个可选的方法被实现而这个方法又并没有被实现的时候，就会抛出一个

一个UnsupportedOperationException异常

。UnsupportedOperationException类继承了RuntimeException类。这使得调用者能够调用所有的集合操作而不需要把每次调用都放在一个try-catch对里。100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com