泛型：

一、什么叫泛型

类型参数化，在定义类时指定一个泛型，然后在创建对象时

将具体的类型传入，目的是为了保证编译期间的类型安全检测机制

泛型只是编译期间，在运行期间就会擦除所有泛型，变成Object类型。

二、泛型的通配符

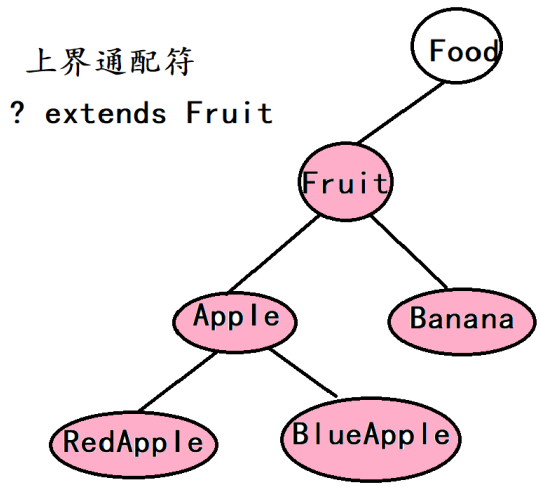
1、上界通配符

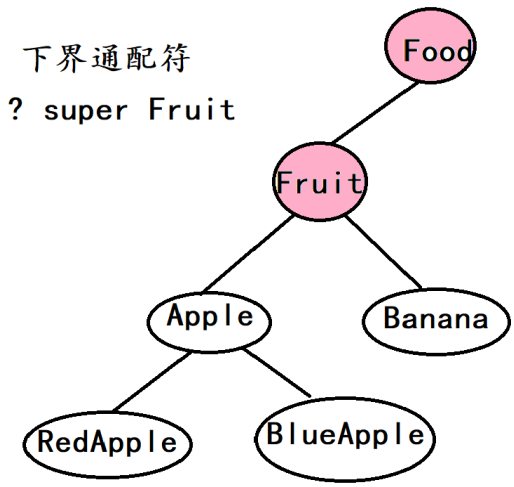
？ extends 类型

类型<? extends T> 是所有<? extends T>和<? extends T的子类>的父类型

2、下界通配符

？ super 类型





冯诺依曼计算机体系结构

1、输入

2、输出

3、控制器

4、运算器

5、存储器

集合

1、软件（程序）的概念

软件 = 数据结构 + 算法。

既： 存数据 + 操作数据

2、数据结构

数据结构本质上是一种容器，可以将内存中一堆数据
按照一定的结构组织并存取起来

常用的数据结构：
1）数组

2）栈： 一端开口，一端闭口。
FILO:先进后出 LIFO:后进先出
存数据的过程叫：压栈 取出的过程叫：弹栈

3）队列： 两端都是开口。
分为队首和队尾，数据顺序为从队尾进去，从队首出去，既只能一个方向进出
FIFO先进先出 LILO后进后出

4）链表 分单向、双向 单向循环、双向循环

5）树
二叉树 左边的一定比父节点小，右边的一定比父节点大。

6）哈希表（散列表）
7）堆
8）图

3、算法
1>算法概念
以什么样的方式去操作数据结构中所存储的一堆数据
2>算法
检索，排序，插入，删除，更新...

集合
一、什么是集合
集合就是数据结构具体的实现方案
二、集合的分类
Collection(父接口)集合 主要用于存储单个元素
子接口
1--->List 列表 能保证元素的添加顺序，允许添加重复元素
1)------->ArrayList(实现类) 底层实现为数组
2)------->LinkedList(实现类) 底层实现为链表

2--->Set 不能保证元素的添加顺序，不允许添加重复元素

1)------->HashSet(实现类) 底层实现为HashMap

2)------->TreeSet(实现类) 底层实现为TreeMap

3--->Queue 队列 底层以单项队列形式来存储元素

1)------->Deque(子接口) 底层以双向队列形式来存储元素

2)------->LinkedList(实现类)

通过链表实现了单项队列，双向队列和栈结构

Map(父接口)映射

key键-value值 键值对。

主要用于存储键值对形式的元素

-->HashMap(实现类) 底层实现为哈希表

-->TreeMap(实现类) 底层为红黑树

集合操作工具

1)Collections用于操作Collection的工具类

2)Iterator 名词 迭代器 用来循环 (Iterat)

3)Comparator 名词 比较器

4)Comparable 形容词 可比较的

5)Arrays

6)Stream 流式操作

三、List接口

列表集合，Collection接口的子接口，主要用于存储单个元素，

存储在该集合中的元素是可以重复的，也能保证元素的添加顺序，

主要实现类：ArrayList,LinkedList

1、ArrayList

1>概念

数组列表集合，底层是将数据存储到数组中

2>对象创建

ArrayList<T> list = new ArrayList<>;

3>主要方法

1)boolean add(E e) 添加一个元素到末尾

2)void add(int index,E e) 向指定下标位置插入一个元素

3)int size() 获取集合长度

4)E get(int index) 根据下标获取一个元素

5)E remove(int index) 根据下标删除一个元素

6)boolean remove(E e) 根据内容删除一个元素

7)E set(int index,E e) 根据下标修改一个元素

8)boolean contains(E e) 判断是否包含指定元素

9)void clear() 清除集合所有内容

10)boolean isEmpty() 判断集合是否为空

11)boolean addAll(Collection c) 向一个集合中添加一个集合

12)boolean removeAll(Collection c) 向一个集合中删除另一个集合

4>List集合三种遍历方式

1)通过下标遍历

注意：如果要删除元素，从后往前倒叙遍历删除

2)for~each

底层调用的是Iterator迭代器

3)Iterator迭代器

3.1)Iterator<E> it = 集合对象.iterator();

获取迭代器对象

3.2)it.hasNext()

判断是否有下一个元素

3.3)it.next()

将指针移动到下一个位置并取当前指向的元素

3.4)it.remove()

通过迭代器删除集合中元素

3.5)快速失败机制

当集合中的元素正在迭代器中进行迭代时，

不能通过集合对元素进行删除和添加操作，

会引发 并发修改异常

只能通过迭代器对象进行删除

参考ArrayList源码，自己实现一个ArrayList

实现添加，插入，删除，查询，修改。

5>ArrayList扩容原理

1)当调用无参构造方法创建对象时，底层会自动创建一共长度为0的Object[]数组

2)当第一次调用add方法时，会将Object[]数组长度扩容到10个。

3)当后续继续调用add()方法时，会先判断容量是否已满，如果数组容量已满，就将Object[]数组扩容到之前长度的1.5倍。

6>Vector类（了解）

1)Vector和ArrayList都是List接口的实现类，底层都是通过Object[]数组存储元素。

2)Vector在通过无参构造方法创建对象时，默认会初始化一个长度为10的Object[]数组，后续需要扩容时，将扩容之前长度的2倍（立即加载）。

3)Vector是线程安全的。

ArrayList是线程不安全的。

7>Stack(栈)

代表栈结构，是继承自Vector的子类，底层是通过Object[]数组来实现的，栈结构遵循FILI先进后出原则。

主要方法：

1、push(E e) 压栈，入栈

2、pop() 弹栈，弹栈

3、peek() 获取但不移除栈顶元素

4、isEmpty() 判断是否为空栈

四、集合操作工具类

1、Comparable

可比较接口

作用：用于比较两个引用类型对象之间的大小

使用方式：

1)被比较的对象类型实现Comparable可比较接口

2)在类中重写compareTo()方法，在方法中自己指定比较规则

3)该类的对象之间就可以调用compareTo()方法来比较大小了

2、Collections

主要用于对Collection接口下的各种集合元素进行操作

主要方法

1)addAll()

一次性向集合中添加多个元素

2)max()

获取集合中最大元素

3)min()

获取集合中最小元素

4)reverse()

翻转集合中元素

5)suffle

打乱集合中元素

6)sort(List<T> list)

排序，对集合中元素进行排序，要求集合中元素类型必须实现Comparable可比较接口，然后按照重写的compareTo方法。

来对元素进行排序

7)sort(List<T> list,Comparator<? super T>c)

对集合中元素按照Comparator比较器制定的规则来进行排序，此时集合中元素类型可以不要实现Comparator可比较接口，即使实现了该接口，也是以Comparator比较规则为准。

Comparable:可比较的

--compareTo(T t)

Comparator:比较器接口

--compare(T t1,T t2)

双向链表 LinkedList

1、LinkedList概念

底层事项为一个双向链表，链表上的每个结点

都可以分为三个部分

1)前驱指针

保存上一个节点的内存地址，指向上一个节点

2)内容

往链表中添加的元素内容

3)后驱指针

保存下一个结点的内存地址，可以找到下一个结点

通过双向链表写队列、双向队列、栈

2、LinkedList作为双向链表使用

实现了List接口，拥有和ArrayList相同的方法，但是方法内部都是通过双向链表来实现的。

3、LinkedList作为单向队列使用

LinkedList实现了Queue接口，所以可以作为单项队列来使用
只要控制双向链表从一端插入数据（队尾），
从另外一端取出数据（队首），就可以通过
双向链表来实现单向队列结构了，
队列遵循FIFO先进先出原则

作为单项队列的主要方法：
1)offer() 队尾入队
2)poll() 队首出队

4、LinkedList作为双向队列使用

LinkedList实现了Deque接口，所以可以作为双向队列来使用
只需要控制允许从双向链表的量短添加和移除数据，就可以通过
双向链表来实现双向队列结构，双向队列也遵循FIFO先进先出原则

作为双向队列的主要方法
1)OfferFirst() 队首入队
2)offerLast() 队尾入队
3)pollFirst() 队首出队
4pollLast() 队尾出队

5、LinkedList作为栈使用

如果控制只允许从LinkedList双向链表的一端添加和删除数据
就可以通过双向链表来实现栈结构了，栈结构遵循FIFO先进先出原则

作为栈的主要方法：
1)push() 压栈，入栈
2)pop() 弹栈，出栈

6、ArrayList 和 LinkedList有什么区别?

1、都实现了Lst接口，因此都能保证元素的添加顺序，

以及可以添加重复怨怒是和null值

2、ArrayList底层实现为数组，因此查找和修改元素比较快

因为可以直接通过下标来一次定位元素

但是删除和插入比较慢，因为需要赋值和移动元素，或扩容

3、LinkedList底层实现为双向链表，因此删除和插入比较快

因为只需要修改结点的前驱和后驱指针，不需要移动赋值元素

但是查找和修改元素比较慢，因为需要从头结点开始逐个向后遍历

或从为结点开始逐个向前遍历