增强型，for循环

使用增强型for循环遍历集合 – Java5.0出现的新特性，该循环不通用于传统for循环工作，其知识用于集合或者数组的遍历

语法：for（元素类型 元素名：集合或者数组名）{

}

增强型for循环不是新语种，知识在编译器在编译生成.class文件时，将新循环转成迭代器，所以在遍历集合时，不能通过集合的方式来改变集合

增强型for循环的本质为迭代器。

注意点：不允许在遍历的过程中修改集合元素，因为本质是迭代器

泛型：

1. 主要作用是限制集合中所存放的元素的类型，方便后期对集合数据的操作，所有集合都支持泛型，迭代器也支持泛型
2. Jdk1.5之后引入的概念

<Integer> 🡪泛型

List 集合：

1. 是可以通过元素下标来操作元素（下标是从0开始）
2. ArrayList：数组结构—检索数组比较快
3. LinkedList：链表结构—插入和删除速度快
4. 在处理数据量比较大的时候，体现的较为明显
5. subList(int index, int endIndex)如果子集发生改变，对应的原集合也会发生改变
6. 集合与数组之间的转换
   1. toArray()：自动转成Object类型的数组，比较麻烦
   2. toArray(new 引用类型[数组长度])：这种转换方式会直接转成我们想要的类型的数组
      1. 数组长度：定义的数组长度，大于集合元素的个数，多余的部分，补充null，
      2. 定义的数组长度，小于集合元素的个数，系统会给我们定义一个默认为集合个数的数组长度，不再使用我们自己定义的数组长度
7. 数组转集合：一般是转成List集合（set中不允许重复）
   1. asList，转成对应泛型的List集合
   2. 从数组转成集合后，不能向集合中添加元素，否则会报异常，需要重新建立一个集合，将该转成后得集合复制过去，就可以想新集合中添加元素
   3. 使用addAll()使用创建集合的有参数的构造方法

Collections：是关于一个Collection接口集合的一个类（工具类），里面封装着一些对集合操作的特定的方法

Sort(List<T> list)：按照默认的排序顺序，给给定的集合进行排序

1. 比较器
   1. 排序接口：comparable，相当“内部比较器”，若一个类实现该接口，意味着“该类支持排序“，该类的对象拥有着比较的资格，但是需要重新定义比较规则，即重写compareTo()方法
      1. 该方法用于使当前对象与给定的对象进行比较，返回的值为int类型的值，与值得大小无关，与值得正负有关
      2. 若当前对象大于给定得对象，返回值大于0 降序
      3. 若当前对象小于给定得对象，返回值小于0 升序
      4. 若当前对象等于给定得对象，返回值为0，顺序不变
   2. 接口comparator比较器，相当于“外部比较器“，不用修改源代码，而是实现另外一个比较器；当对象需要比较时，将对象集合和比较器一起传过去即可
   3. 匿名内部类
      1. Comparator<String> c = new Comparator<String>{};
   4. 什么时候使用外部比较器：
      1. 一般情况下，是当集合中的元素，已经实现了comparator接口，重写了比较规则，但是该比较规则满足不了业务需求，此时我们需要定义一个外部比较器，按照业务需求定义比较规则，如果只是想使用一次，则最好定义成匿名内部类
2. Collection的子接口：
   1. Set接口：无序不重复（添加重复元素会失败）
   2. Set<引用类型> set = new hashSet<引用类型>

队列：是一种特殊的线性表，它限制了对线性表的访问从一端添入，另一端取出

Poll 取出元素，队列中也出队的操作

Peek只能取队首的引用，队列中依旧存在的，不能进行出队操作

队列只能进行一次操作，操作之后，队列中就不存在元素了Queue

栈：其实也叫双端队列，用来记录一组可追溯的数据，计算机当中的栈的原理就是使用队列做的

原则：先进后出，每次操作只能栈顶元素 Deque

入栈push，，，出栈只能从栈顶pop，，，

栈的遍历也是一次性的，只能遍历一次，