看看数据集中的某些元素是否匹配一个给定的属性。Stream API通过allMatch、anyMatch、noneMatch、findFirst和findAny方法提供了这样的工具。

#### 1、检查谓词是否至少匹配一个元素（anyMatch）

anyMatch方法可以返回“流中是否有一个元素能匹配给定的谓词”。可以用它来看看人员里是否有胖子：

if (list.stream().anyMatch(Person::*isFat*)) {

System.*out*.println("The people is fat!");  
 return true;  
}

#### 2、检查谓词是否匹配所有元素（allMatch、noneMatch）

allMatch看看流中的元素是否都能匹配给定的谓词。比如是否所有人员的体重都小于80kg：

boolean isHealthy = list.stream().anyMatch(p -> p.getWeigth() < 80);

noneMatch：它可以确保流中没有任何元素与给定的谓词匹配。

anyMatch、allMatch和noneMatch这3个操作都用到了所谓的短路，就是大家熟悉的Java中&&和||运算符短路在流中的版本。

短路：有些操作不需要处理整个流就能得到结果。例如，假设你需要对一个用and连起来的大布 尔表达式求值。不管表达式有多长，你只需找到一个表达式为false，就可以推断整个表达式 将返回false，所以用不着计算整个表达式。

#### 3、查找元素

findAny方法将返回当前流中的任意元素。可以与其他流操作结合使用。比如，你可能想找到一名偏胖的人员。结合filter和findAny方法来实现这个查询：

Optional<Person> person = list.stream()

.filter(Person::*isFat*)  
 .findAny();

流水线将在后台进行优化使其只需走一遍，并在利用短路找到结果时立即结束。

Optional简介

Optional<T>类（java.util.Optional）是一个容器类，代表一个值存在或不存在。findAny可能什么元素都找不到。Java 8引入Optional<T>，这样就不用返回容易出问题的null了。该类里有几种可以迫使你显式地检查值是否存在或处理值不存在的情形的方法。

isPresent()：Optional包含值返回true,否则返回false。

isPresent(Consumer<T> block)：在值存在的时候执行指定代码块。