

内存资源分配题目描述：

有一个简易内存池，内存按照大小粒度分类，每个粒度有若干个可用内存资源，用户会进行一系列内存申请，需要按需分配内存池中的资源，返回申请结果成功失败列表。分配规则如下：

- 1、分配的内存要大于等于内存申请量，存在满足需求的内存就必须分配，优先分配粒度小的，但内存不能拆分使用。
- 2、需要按申请顺序分配，先申请的先分配。
- 3、有可用内存分配则申请结果为 *true*，没有可用内存分配则返回 *false*。

注：不考虑内存释放。

输入描述：

输入为两行字符串：

第一行为内存池资源列表，包含内存粒度数据信息，粒度数据间用逗号分割，一个粒度信息内部用冒号分割，冒号前为内存粒度大小，冒号后为数量。资源列表不大于 *1024*，每个粒度的数量不大于 *4096*

第二行为申请列表，申请的内存大小间用逗号分隔。申请列表不大于 *100000*

如：

64:2,128:1,32:4,1:128

50,36,64,128,127

输出描述：

输出为内存池分配结果。

如：

true,true,true,false,false

示例 1

输入：

64:2,128:1,32:4,1:128

50,36,64,128,127

输出:

true,true,true,false,false

说明:

内存池资源包含: 64K 共 2 个、128K 共 1 个、32K 共 4 个、1K 共 128 个的内存资源;

针对 50,36,64,128,127 的内存申请序列, 分配的内存依次是:

64,64,128,NULL,NULL,第三次申请内存时已经将 128 分配出去,

因此输出结果是: true,true,true,false,false

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import java.util.Scanner;
```

```
import java.util.TreeMap;
```

```
// 注意类名必须为 Main, 不要有任何 package xxx 信息
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in, "UTF-8");
```

```
        // 输入处理: 输入 2 行字符串
```

```
        String strFirst = scanner.nextLine();
```

```
        String strSecond = scanner.nextLine();
```

```
        // 进行逗号分隔
```

```
        String[] split = strFirst.split(",");
```

```
        String[] asks = strSecond.split(",");
```

```
        // 初始化内存
```

```
        TreeMap<Integer, Integer> map = new TreeMap<>();
```

```
        for (String s : split) {
```

```
            Integer size = Integer.parseInt(s.split(":")[0]);
```

```
            Integer num = Integer.parseInt(s.split(":")[1]);
```

```
            map.put(size, num);
```

```
        }
```

```
        // 结果输出
```

```
        StringBuilder sb = new StringBuilder();
```

```
        ArrayList<Integer> list = new ArrayList<>(map.keySet());
```

```
        for (String ask : asks) {
```

```
            int flag = 0;
```

```
        for (Integer integer : list) {
            int askInt = Integer.parseInt(ask);
            if (askInt <= integer && map.get(integer) > 0) {
                sb.append("true").append(",");
                map.put(integer, map.get(integer) - 1);
                flag = 1;
                break;
            }
        }
        if (flag == 0) {
            sb.append("false").append(",");
        }
    }
    System.out.println(sb.substring(0, sb.length() - 1));
}
```