

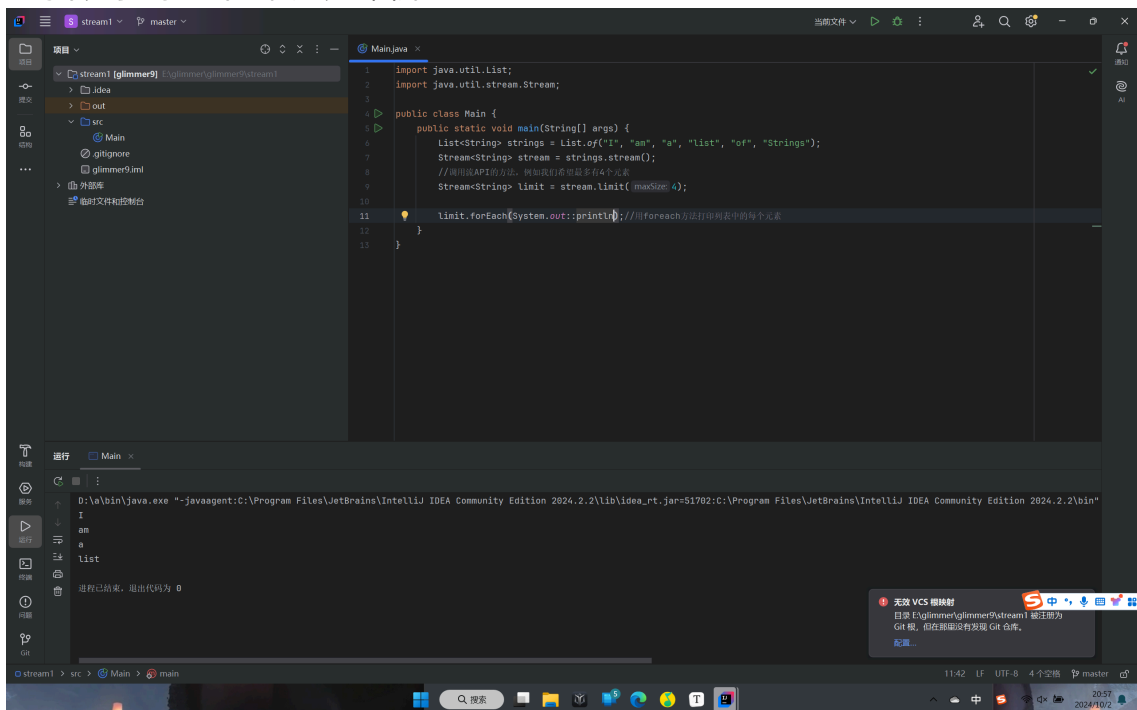
对09的回答

具体答题思路参见注释，如果觉得思路不够清晰，欢迎随时拷打--

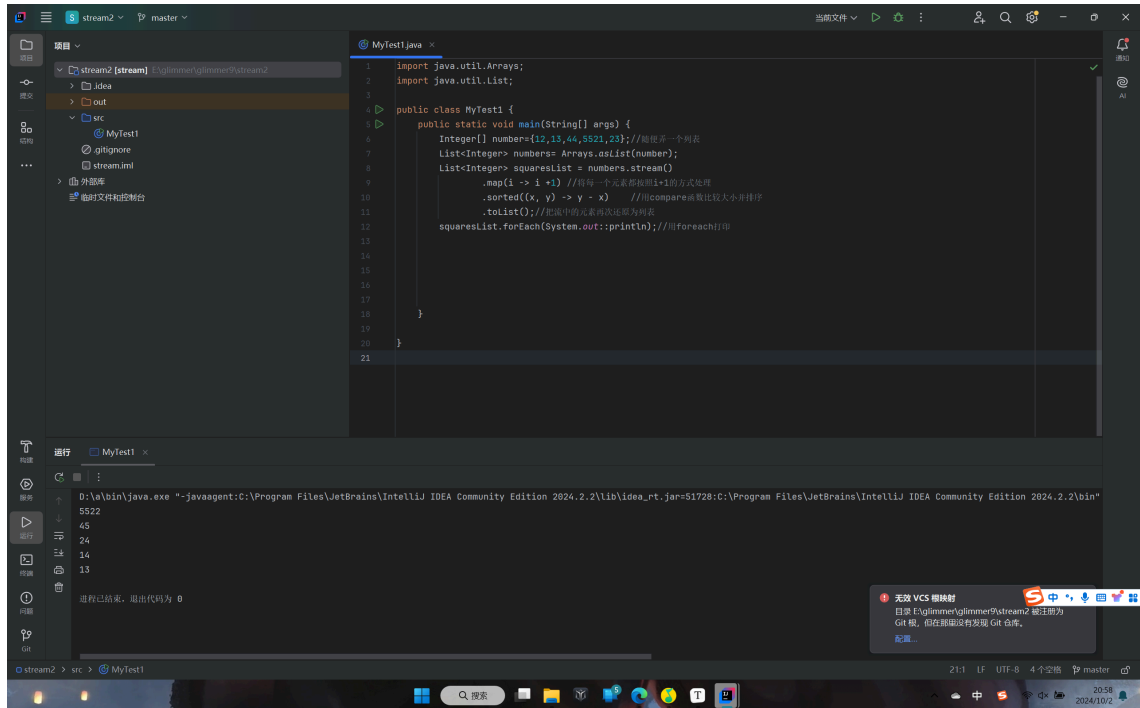
1. 关于对第一段代码的修改详情见stream1

```
List<String> strings = List.of("I", "am", "a", "list", "of", "Strings");
Stream<String> stream = strings.stream();
//调用流API的方法，例如我们希望最多有4个元素
Stream<String> limit = stream.limit(4);
//最后我们打印结果
System.out.println("limit = " + limit);//list直接打印会按照toString方法，会打一个地址加哈希值
```

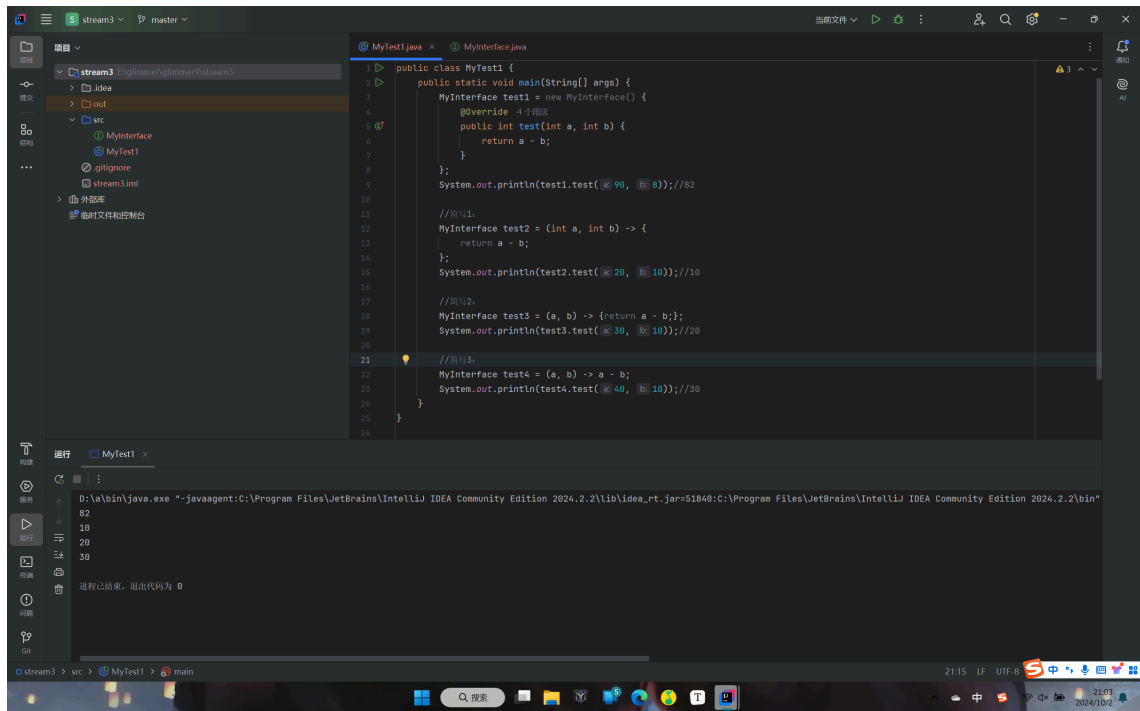
可以通过重载toString方法实现，但为了方便，可以直接调用list类中的foreach方法直接遍历list中的每个元素。挂一个运行成功的图片



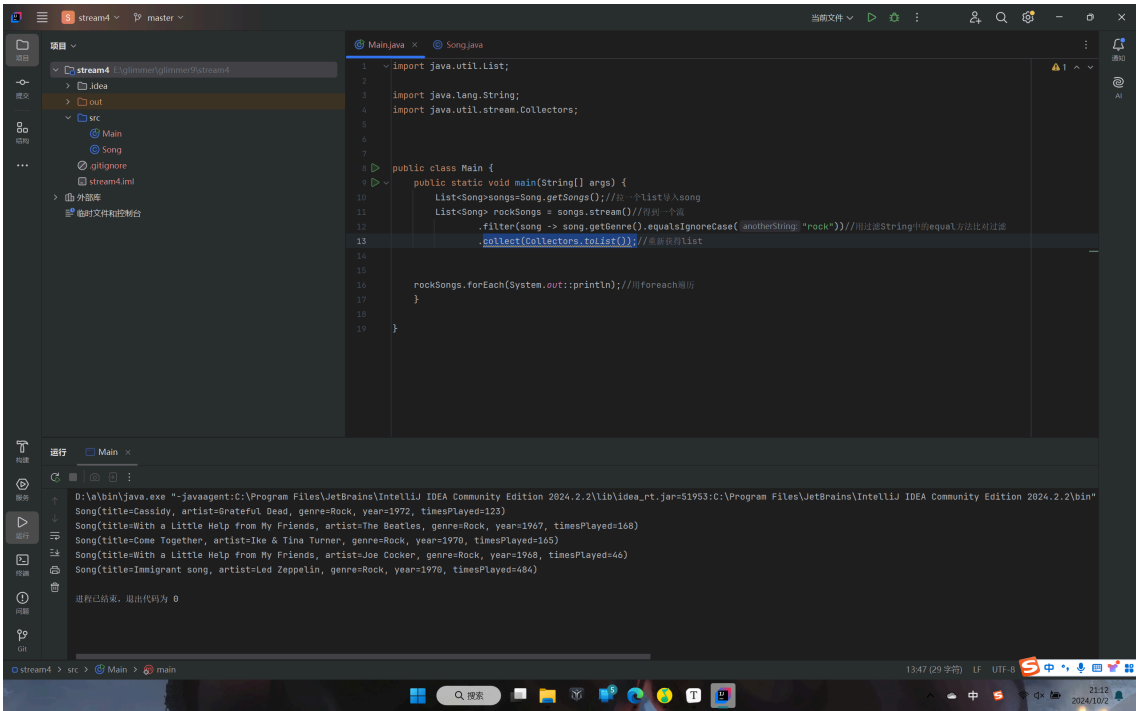
2. 对于第二段stream流的应用详见stream2，我写了注释，挂一个成功的截图



3. stream3是我自己捣鼓lambda语法的时候留下的，其实就是把题目给的链接里的例子自己随便写了写，让他能跑，同样的放一张图



4. stream4是筛选rock的实现，那个tolist网上查了查似乎只在部分编译器内能用，而 collect(Collectors.toList());是通用的，所以写成了后一种，上图



```
import java.util.List;
import java.lang.String;
import java.util.stream.Collectors;

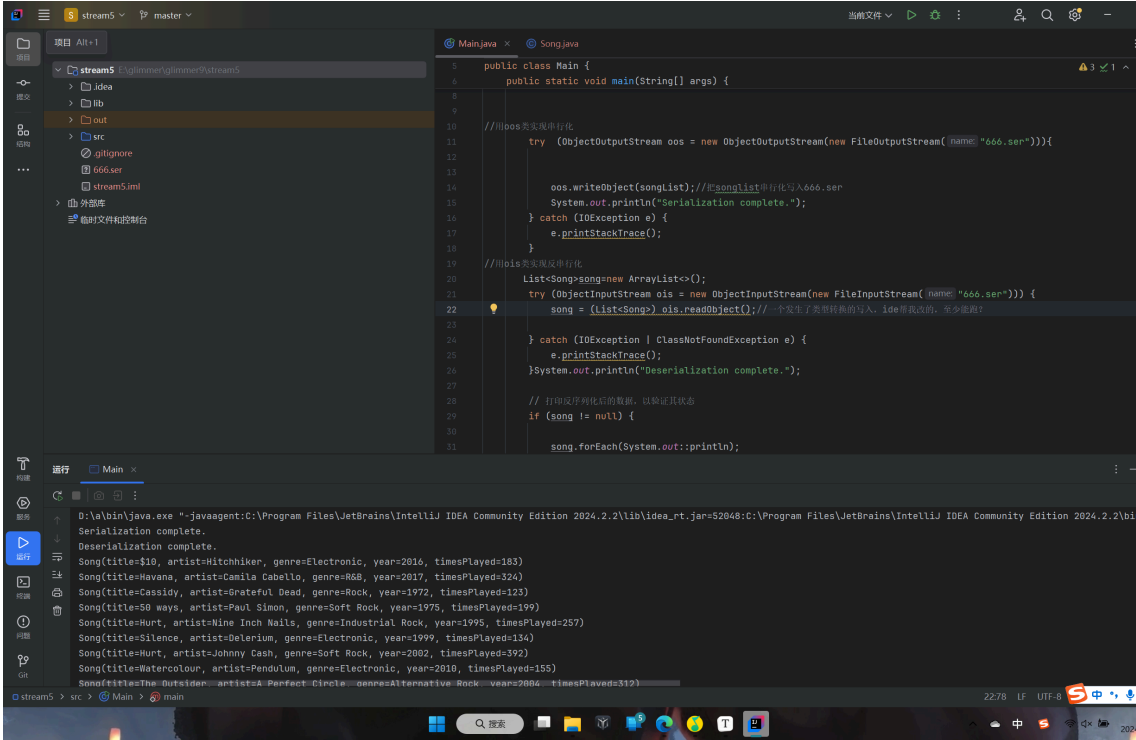
public class Main {
    public static void main(String[] args) {
        List<Song> songs = Song.getSongs(); // 这是一个List<Song>
        List<Song> rockSongs = songs.stream().filter(song -> song.getGenre().equalsIgnoreCase("rock")).collect(Collectors.toList()); // 通过String中的equal方法比对过滤
        // 返回新的List

        rockSongs.forEach(System.out::println); // 用foreach遍历
    }
}
```

运行结果:

```
D:\a\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\lib\idea_rt.jar=51953:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\bin"
Song(title=Cassidy, artist=Grateful Dead, genre=Rock, year=1972, timesPlayed=123)
Song(title=With a Little Help from My Friends, artist=The Beatles, genre=Rock, year=1967, timesPlayed=168)
Song(title=Come Together, artist=Ike & Tina Turner, genre=Rock, year=1970, timesPlayed=165)
Song(title=With a Little Help from My Friends, artist=Joe Cocker, genre=Rock, year=1968, timesPlayed=46)
Song(title=Immigrant song, artist=Led Zeppelin, genre=Rock, year=1970, timesPlayed=486)
```

5. stream5实现了串行化与反串行化，有一处神奇的类型转换，感觉有点怪，但能跑，



```
public class Main {
    public static void main(String[] args) {
        // 用Oos类实现串行化
        try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("666.ser"))){
            oos.writeObject(songList); // 将songList序列化写入666.ser
            System.out.println("Serialization complete.");
        } catch (IOException e) {
            e.printStackTrace();
        }

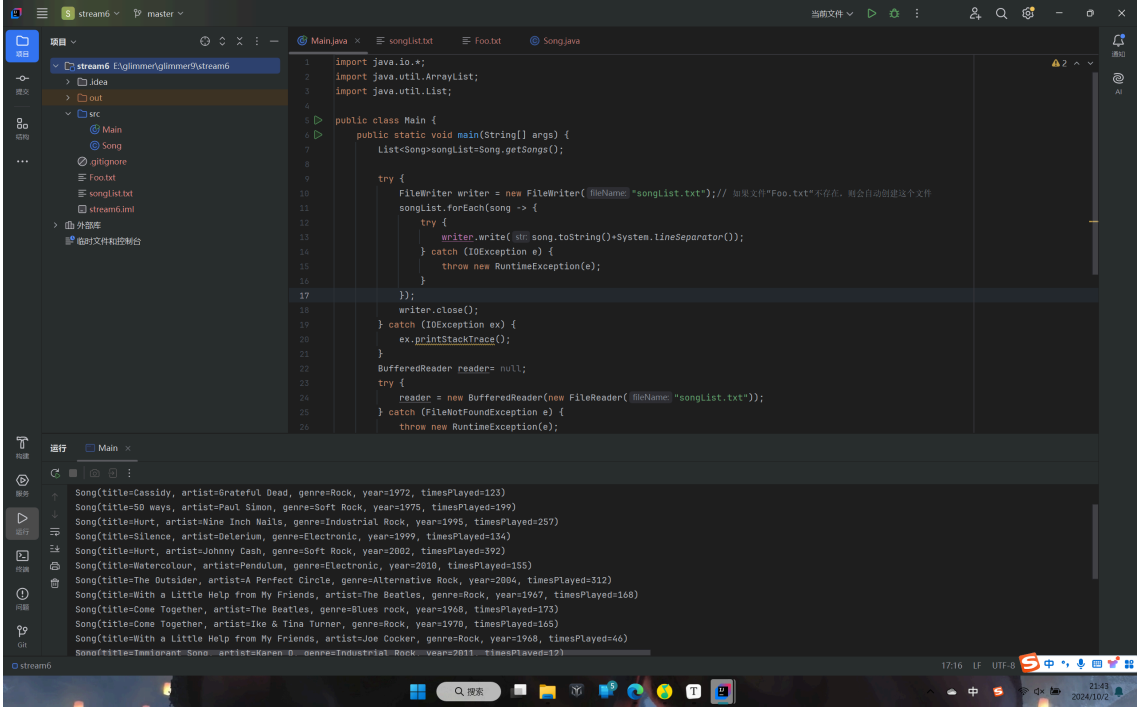
        // 用Ois类实现反串行化
        List<Song> songs = new ArrayList<>();
        try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream("666.ser"))){
            song = (List<Song>) ois.readObject(); // 一个发生了类型转换的写入，ide帮我改的，至少能跑！
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
        System.out.println("Deserialization complete.");

        // 打印反序列化后的数据，以验证其状态
        if (song != null) {
            song.forEach(System.out::println);
        }
    }
}
```

运行结果:

```
D:\a\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\lib\idea_rt.jar=52048:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.2\bin"
Serialization complete.
Deserialization complete.
Song(title=$10, artist=Hitchhiker, genre=Electronic, year=2016, timesPlayed=183)
Song(title=Havana, artist=Camila Cabello, genre=R&B, year=2017, timesPlayed=324)
Song(title=Cassidy, artist=Grateful Dead, genre=Rock, year=1972, timesPlayed=123)
Song(title=50 ways, artist=Paul Simon, genre=Soft Rock, year=1975, timesPlayed=199)
Song(title=Hurt, artist=Mine Inoh Moits, genre=Industrial Rock, year=1995, timesPlayed=257)
Song(title=Silence, artist=Botanum, genre=Electronic, year=1999, timesPlayed=134)
Song(title=Hurt, artist=Johnny Cash, genre=Soft Rock, year=2002, timesPlayed=392)
Song(title=Watercolour, artist=Pendulum, genre=Electronic, year=2010, timesPlayed=155)
Song(title=The Outside, artist=The Perfect Circle, genre=Alternative Rock, year=2004, timesPlayed=512)
```

6. stream6实现了文件的读写，其实大佬，你是否还记得，文件的I/O在07里就写过，



写了整整一天啊，痛苦面具，大佬给个高分吧