Java07

Task1

Q1:

List类的集合都：有序，可重复，有索引。

Set类的集合都：不重复，无索引。LinkedHashSet有序，HashSet无序，TreeSet自动排序。

Map集合是双值集合，元素都是键值对。

集合包含数组，数组包含于集合，数组只能装数，集合可以装任何东西（object）

Task2

Q2：以下是截图

```
package task7;

import java.util.ArrayList;
import java.util.List;

public class Task7_Q2 {
    public static void main(String[] args) {
        List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
        list.add(4);

        for (int i : list) {
            System.out.println(i);
        }
    }
}
```

代码附上~~~

Q3:

匿名内部类就是创建了一个子类，这个子类没有名字（匿名了）

函数式接口就是只有一个抽象方法的接口，只有函数式接口才可以使用lambda表达式

以下是lambda表达式的改写方法：

```
list.forEach((integer) -> System.out.println(integer));
```

本题目要求简化匿名内部类，创建一个lambda表达式，做法为：

只保留lambda表达式的参数，去掉参数类型，保留方法体，去掉return，若处理后只有一行，还可以去掉{}及其分号

Task3

Q4：

这是接口

```
package task7_Q4;

public interface Repository {
    void save(int id, User object);
    User findById(String id);
}
```

这是用户类

```
package task7_Q4;

public class User {
    private String name;
    private int age;

    public User(String name, int age) {
        this.name = name;
        this.age = age;
    }

    @Override
    public String toString() {
        return "User{name='" + name + "', age=" + age + "}";
    }
}
```

这是实现类

```
package task7_Q4;

import java.util.HashMap;
import java.util.Set;

public class MyRepository implements Repository{
    HashMap<Integer,User> map = new HashMap();
```

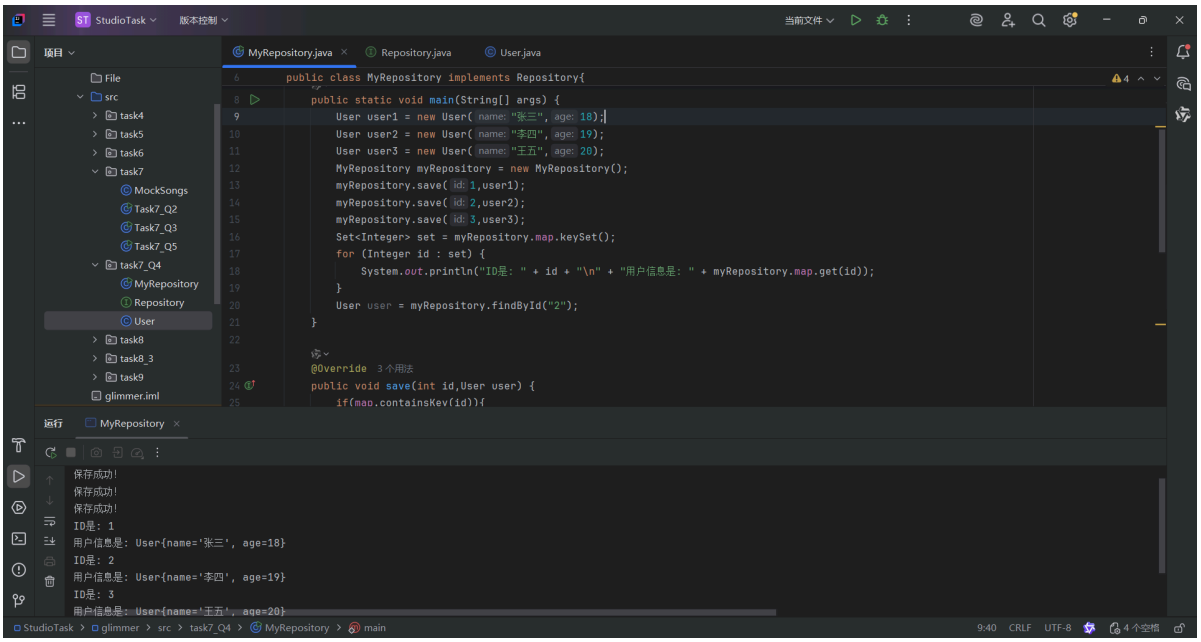
```
    public static void main(String[] args) {
        User user1 = new User("张三",18);
        User user2 = new User("李四",19);
        User user3 = new User("王五",20);
        MyRepository myRepository = new MyRepository();
        myRepository.save(1,user1);
        myRepository.save(2,user2);
        myRepository.save(3,user3);
        Set<Integer> set = myRepository.map.keySet();
        for (Integer id : set) {
            System.out.println("ID是: " + id + "\n" + "用户信息是: " +
myRepository.map.get(id));
        }
        User user = myRepository.findById("2");
    }

    @Override
    public void save(int id,User user) {
        if(map.containsKey(id)){
            System.out.println("输入ID已存在，请重新输入！");
        }else{
            map.put(id,user);
            System.out.println("保存成功！");
        }
    }

    @Override
    public User findById(String id) {
        if(map.containsKey(Integer.valueOf(id))){
            return map.get(id);
        }else {
            System.out.println("未找到该用户！");
            return null;
        }
    }
}
```

遍历方法使用的是键找值方法：先把键放在一个Set集合，再遍历这个Set集合

以下是代码的运行截图



Task4

以下是测试类

```
import java.util.List;

public class Task7_Q5 {
    public static void main(String[] args) {
        List list = MockSongs.getSongStrings();
        System.out.println(list);
    }
}
```

这是方法的内容

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;

public class MockSongs {
    public static List<String> getSongStrings(){
        List<String> songs = new ArrayList<>();
        //模拟将要处理的列表
        songs.add("sunrise");
        songs.add("thanks");
        songs.add("$100");
        songs.add("havana");
        songs.add("114514");
        //TODO
        sort( songs);
        //END
        return songs;
    }
```

```
    //重写排序方法
    public static void sort(List<String> songs){
        Collections.sort(songs, new Comparator<String>() {
            @Override
            public int compare(String o1, String o2) {
                if (o1.length() > o2.length()) {
                    return 1;
                } else if (o1.length() < o2.length()) {
                    return -1;
                }else {
                    return o2.compareTo(o1);
                }
            }
        });
    }
}
```

排序方法使用的是匿名内部类

本题还涉及特殊字符排序的处理，我本来一头雾水，但是注意到特殊字符的长度是独一无二的，所以可以利用这个特性来绕过它的比较
谢大佬的不杀之恩！！！（送花）