抽象クラスの抽出

オブジェクト指向プログラミング特論

2020年度

只木進一：理工学研究科

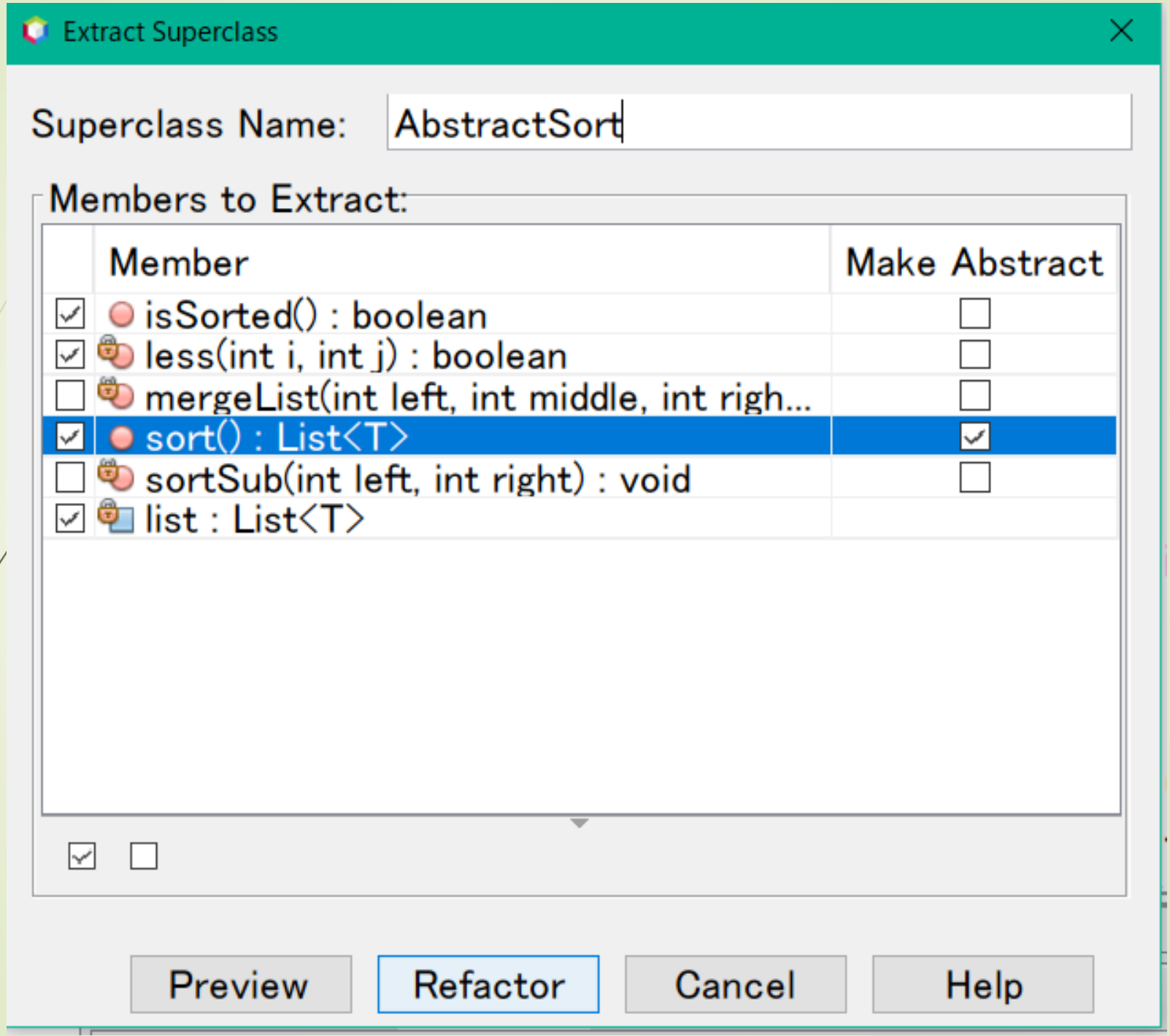
抽象クラスの抽出

- 既存のクラスから共通的部分を抽出
- Netbeansでは、「リファクタリング」機能で可能

- example2パッケージへコピー
  - BubbleSort
  - MergeSort
  - import example1.*を削除

MergeSortからの抽出

- そのまま抽出
  - less()、isSorted()、list
- 抽象化して抽出
  - sort()
- AbstractSortクラスとして保存
- コンストラクタに注意


Extract Superclass
Superclass Name:
AbstractSort
Members to Extract:
Member
Make Abstract
isSorted() : boolean
less(int i, int j) : boolean
mergeList(int left, int middle, int righ...
sort() : List<T>
sortSub(int left, int right) : void
list : List<T>
Preview
Refactor
Cancel
Help

5

```
import java.util.List;
import jdk.internal.HotSpotIntrinsicCandidate;

/**
 *
 * @author tadaki
 */
public abstract class AbstractSort<T extends Comparable<T>> {

    protected final List<T> list;

    @HotSpotIntrinsicCandidate
    public AbstractSort(List<T> list) {
        this.list=list;
    }
```

削除

削除

適切に直す

```java
/**
 * MergeSortの基本的実装
 *
 * @author tadaki
 */
public class MergeSort<T extends Comparable<T>> extends AbstractSort<T>{


    public MergeSort(List<T> list) {
        super(list);
    }

    /**
     * 整列の実行
     *
     * @return 整列済みのリスト
     */
    public List<T> sort() {
        sortSub(0, list.size());
        return list;
    }
```

適切に直す

AbstractSortの継承クラス

- MergeSort
  - sort()を上書き
- BubbleSort
  - sort()を上書き
  - less()、swap()、isSorted()をAbstractSortへ

```
 * @param <T>
 */
public class BubbleSort<T extends Comparable<T>>  extends AbstractSort{

    //final private List<T> list;

    public BubbleSort(List<T> list) {
        super(list);
    }

    /**
     * 整列の実行
     *
     * @return 整列済みのリスト
     */
    @Override
    public List<T> sort() {
```

削除

課題：SelectionSortを継承クラスとして作成

n //要素数

for ($i = 0$; $i < n - 1$; $i++$){

  m = ($i < j < n$の範囲の最小要素の位置)

  if ($i \neq m$)swap(i, m)

}