1. Here is an example from org.eclipse.compare.contentmergeviewer.TexMergeViewer between v20060605 and v20060917.

```
public class TextMergeViewer {
    private HashMap fNewAncestorRanges = new HashMap();
    private HashMap fNewLeftRanges = new HashMap();
    private HashMap fNewRightRanges = new HashMap();

    private Position getNewRange(char type, Object input) {
    switch (type) {
    - case 'A':
    + case ANCESTOR_CONTRIBUTOR:
    return (Position)fNewAncestorRanges.get(input);
    - case 'L':
    + case LEFT_CONTRIBUTOR:
    return (Position)fNewLeftRanges.get(input);
    - case 'R':
    + case RIGHT_CONTRIBUTOR:
    return (Position)fNewRightRanges.get(input);
    }
    return null;
    }

    class Diff {
        Position fAncestorPos;
        Position fLeftPos;
        Position fRightPos;

        private Position getPosition(char type){
          switch (type) {
            - case 'A':
            + case ANCESTOR_CONTRIBUTOR:
              return fAncestorPos;
            - case 'L':
            + case LEFT_CONTRIBUTOR:
              return fLeftPos;
            - case 'R':
```

```
            + case RIGHT_CONTRIBUTOR:
             return fRightPos;
          }
        return null;
      }
    }
  }
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common change extraction refactoring.

```
public class TextMergeViewer {
    private HashMap fNewAncestorRanges = new HashMap();
    private HashMap fNewLeftRanges = new HashMap();
    private HashMap fNewRightRanges = new HashMap();
    private Position getNewRange(char type, Object input) {
        ReturnObj returnObj = extractMethod(type,
                (Position) fNewAncestorRanges.get(input),
                (Position) fNewLeftRanges.get(input),
                (Position) fNewRightRanges.get(input));
        if (returnObj.type == RETURNTYPE.RETURN)
            return returnObj.value;
        else
            return null;
    }
    class Diff {
        Position fAncestorPos;
        Position fLeftPos;
        Position fRightPos;

        private Position getPosition(char type) {
            ReturnObj returnObj = extractMethod(type, fAncestorPos, fLeftPos, fRightPos);
            if (returnObj.type == RETURNTYPE.RETURN)
                return returnObj.value;
            else
                return null;
        }
    }
    class ReturnObj {
        public RETURNTYPE type;
        public Position value;

        public ReturnObj(RETURNTYPE type, Position value) {
            this.type = type;
            this.value = value;
        }
    }
    enum RETURNTYPE {
        RETURN, FALL
    }
    public ReturnObj extractMethod(char type, Position fNewAncestorRanges,
            Position fNewLeftRanges, Position fNewRightRanges) {
        switch (type) {
        case ANCESTOR_CONTRIBUTOR:
            returnObj = new ReturnObj(RETURNTYPE.RETURN, fNewAncestorRanges);
        case LEFT_CONTRIBUTOR:
            returnObj = new ReturnObj(RETURNTYPE.RETURN, fNewLeftRanges);
        case RIGHT_CONTRIBUTOR:
            returnObj = new ReturnObj(RETURNTYPE.RETURN, fNewRightRanges);
        }
        return new ReturnObj(RETURNTYPE.FALL, null);
    }
}
```

2. Here is an example from org.eclipse.compare.contentmergeviewer.TextMergeViewer.ContributorInfo between v20060918 and v20061016.

```
class ContributorInfo {
    private IEditorInput getDocumentKey() {…}
    private void resetDocument() {…}
    public void elementMoved(Object originalElement, Object movedElement) {
```

```
    + IEditorInput input = getDocumentKey();
    + if(input != null && input.equals(originalElement)){
    + resetDocument();
    +}
    }
     public void elementDeleted(Object element) {
    + IEditorInput input = getDocumentKey();
    + if(input != null && input.equals(element)){
    + resetDocument();
    +}
    }
 }
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
 class ContributorInfo {
     private IEditorInput getDocumentKey() {…}
     private void resetDocument() {…}
     private void extractMethod( Object originalElement) {
         IEditorInput input = getDocumentKey();
         if(input != null && input.equals(originalElement)){
         resetDocument();
     }
     public void elementMoved(Object originalElement, Object movedElement)    {
        extractMethod(originalElement);
     }
    public void elementDeleted(Object element) {
        extractMethod(element);
     }
 }
```

3. Here is an example from org.eclipse.compare.contentmergeviewer.ContentMergeViewer between v20060918 and v20061016.

```
 public class ContentMergeViewer {
     public void flush(IProgressMonitor monitor) {
    - saveContent(getInput());
    + flushContent(getInput(), monitor);
    }
     class SaveAction {
         public void run() {
      - saveContent(getInput());
      + flushContent(getInput(), null);
      }
     }
    public void getInput() {...}
     void flushContent(Object input, IProgressMonitor monitor) {…}
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
 public class ContentMergeViewer {
     public void flush(IProgressMonitor monitor) {
           extractMethod(getInput(), monitor);
    }
     class SaveAction {
         public void run() {
             extractMethod (getInput(), null);
         }
     }
     private void extractMethod (InputObj input, IProgressMonitor monitor monitor ) {
         flushContent(input, monitor);
     }
     void flushContent(Object input, IProgressMonitor monitor) {…}
     public void getInput() {...}
 }
```

4. Here is an example from org.eclipse.compare.CompareEditorInput between v20061120 and v20061218.

```
public class CompareEditorInput {
    private ICompareContainer fContainer;
    public void addCompareInputChangeListener(ICompareInput input,
            ICompareInputChangeListener listener) {
   - if (fContainer == null)
   - input.addCompareInputChangeListener(listener);
   - else
        fContainer.addCompareInputChangeListener(input, listener);
   }
    public void removeCompareInputChangeListener(ICompareInput input,
            ICompareInputChangeListener listener) {
      - if (fContainer == null)
      - input.removeCompareInputChangeListener(listener);
      - else
      fContainer.removeCompareInputChangeListener(input, listener);
    }
}
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class CompareEditorInput {
    private ICompareContainer fContainer;
    public void addCompareInputChangeListener(ICompareInput input,
            ICompareInputChangeListener listener) {
        AddStrategy.compareInputChangeListener(input,listener);
   }
```

```
    public void removeCompareInputChangeListener(ICompareInput input,
            ICompareInputChangeListener listener) {
        RemoveStrategy.compareInputChangeListener(input,listener);
    }
}
abstract class Strategy() {
        public abstract static void compareInputChangeListener(ICompareInput
input,ICompareInputChangeListener listener) ;
}
class RemoveStrategy extends Strategy {
        public void compareInputChangeListener(ICompareInput input,
ICompareInputChangeListener listener) {
            fContainer.removeCompareInputChangeListener(input, listener);
        }
}
class AddStrategy  extends Strategy {
        public void compareInputChangeListener(ICompareInput input,
ICompareInputChangeListener listener) {
            fContainer.addCompareInputChangeListener(input, listener);
        }
}
```

5. Here is an example from org.eclipse.compare.contentmergeviewer. TextMergeViewer.ContributorInfo between v20070226 and v20070316.

```
class ContributorInfo{
   private final TextMergeViewer fViewer;
   private char fLeg;
   private IEditorInput getDocumentKey() {…}
   private IDocumentProvider getDocumentProvider() {…}
   public void elementDirtyStateChanged(Object element, boolean isDirty) {
   + if (!checkState())
   + return;
   IEditorInput input = getDocumentKey();
   if (input != null && input.equals(element)) {
       this.fViewer.updateDirtyState(input, getDocumentProvider(), fLeg);
   }
```

```
    }
    public void elementContentReplaced(Object element) {
    + if (!checkState())
    + return;
    IEditorInput input = getDocumentKey();
    if (input != null && input.equals(element)) {
        this.fViewer.updateDirtyState(input, getDocumentProvider(), fLeg);
    }
    }
  }
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
 class ContributorInfo {
     private final TextMergeViewer fViewer;
     private char fLeg;
     private IEditorInput getDocumentKey() {...}
     private IDocumentProvider getDocumentProvider() {...}
     Label extractMethod() {
         if (!checkState())
             return Label.RETURN;
         return Label.FALL;
     }
     enum Label {
         RETURN, FALL
     }
     public void elementDirtyStateChanged(Object element, boolean isDirty) {
    Label l = extractMethod();
    if (l==Label.RETURN)
     return;
    IEditorInput input = getDocumentKey();
    if (input != null && input.equals(element)) {
        this.fViewer.updateDirtyState(input, getDocumentProvider(), fLeg);
    }   public void elementContentReplaced(Object element) {
        Label l = extractMethod();
        if (l == Label.RETURN)
            return;
        IEditorInput input = getDocumentKey();
        if (input != null && input.equals(element)) {
            this.fViewer.updateDirtyState(input, getDocumentProvider(), fLeg);
        }
    }
}
```

6. Here is an example from org.eclipse.compare.EditionSelectionDialog between v20061016 and v20061030.

```
public class ResizableDialog {
    static void applyDialogFont(Control control){…}
    protected Label statusLabel;
}
public class EditionSelectionDialog extends ResizableDialog {
    private Splitter vsplitter;
    private CompareConfiguration getCompareConfiguration() {…}
    protected synchronized Control createDialogArea(Composite parent2) {
   Composite parent= (Composite) super.createDialogArea(parent2);
   vsplitter.setWeights(new int[] { 30, 70 });
   + IPreferenceStore store= getCompareConfiguration().getPreferenceStore();
   + if (store != null) {
   + if (store.getBoolean(ComparePreferencePage.SHOW MORE INFO)) {
   + statusLabel = new Label(parent, SWT.NONE);
   + statusLabel.setLayoutData(new GridData(
   GridData.FILL HORIZONTAL));
    +}
   +}
   applyDialogFont(parent);
   return parent;
   }
}
public class CompareDialog extends ResizableDialog {
    private Control c;
    private CompareEditorInput fCompareEditorInput;
    protected Control createDialogArea(Composite parent2) {
```

```
    Composite parent= (Composite) super.createDialogArea(parent2);
    c.setLayoutData(new GridData(GridData.FILL BOTH));
    + IPreferenceStore store=
fCompareEditorInput.getCompareConfiguration().getPreferenceStore();
    + if (store != null) {
    + if (store.getBoolean(ComparePreferencePage.SHOW MORE INFO)) {
    + statusLabel = new Label(parent, SWT.NONE);
    + statusLabel.setLayoutData(new GridData(
    GridData.FILL HORIZONTAL));
    +}
    +}
    applyDialogFont(parent);
    return parent;
    }
}
```

We extract common code edits with *extractMethod* refactoring and put the common method in the parent class of both classes. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class ResizableDialog {
    static void applyDialogFont(Control control){…}
    protected Label statusLabel;
    private StatusLabel extractMethod (IPreferenceStore store) {
        if (store != null) {
            if (store.getBoolean(ComparePreferencePage.SHOW MORE INFO)) {
              Label statusLabel = new Label(parent, SWT.NONE);
              statusLabel.setLayoutData(new GridData(GridData.FILL HORIZONTAL));
            }
        }
        return statusLabel;
    }
}
public class EditionSelectionDialog extends ResizableDialog {
    private Splitter vsplitter;
    private CompareConfiguration getCompareConfiguration() {…}
    protected synchronized Control createDialogArea(Composite parent2) {
        Composite parent = (Composite) super.createDialogArea(parent2);
        vsplitter.setWeights(new int[] { 30, 70 });
        statusLabel = extractMethod(getCompareConfiguration().getPreferenceStore());
        applyDialogFont(parent);
        return parent;
    }
}

public class CompareDialog extends ResizableDialog {
    private Control c;
    private CompareEditorInput fCompareEditorInput;
    protected Control createDialogArea(Composite parent2) {
   Composite parent= (Composite) super.createDialogArea(parent2);
   c.setLayoutData(new GridData(GridData.FILL BOTH));
   statusLabel =
extractMethod(fCompareEditorInput.getCompareConfiguration().getPreferenceStore());
   applyDialogFont(parent);
   return parent;
   }
}
```

7. Here is an example from org.eclipse.compare.internal.patch.Patcher-CompareEditorInput between v20061016 and v20061030.

```
public class PatcherCompareEditorInput extends CompareEditorInput {
    protected CompareConfiguration config;
    public CompareConfiguration getCompareConfiguration() {…}
private void initLabels() {
    CompareConfiguration cc = getCompareConfiguration(); cc.setLeftEditable(false);
    cc.setRightEditable(false);
    - String leftLabel = PatchMessages.PatcherCompareEditorInput LocalCopy;
    - cc.setLeftLabel(leftLabel);
    - String rightLabel = PatchMessages.PatcherCompareEditorInput AfterPatch; -
cc.setRightLabel(rightLabel);
    + if (config != null){
```

```
    + cc.setLeftLabel(config.getLeftLabel(config));
    + cc.setLeftImage(config.getLeftImage(config));
    + cc.setRightLabel(config.getRightLabel(config));
+cc.setRightImage(config.getRightImage(config));
    +}else{
    + String leftLabel = PatchMessages.PatcherCompareEditorInput LocalCopy;
    + cc.setLeftLabel(leftLabel);
    + String rightLabel = PatchMessages.PatcherCompareEditorInput AfterPatch;
    + cc.setRightLabel(rightLabel);
    +}
    }
}
public class HunkMergePageInput extends PatcherCompareEditorInput {
    private void initLabels() {CompareConfiguration cc =
getCompareConfiguration();cc.setCalculateDiffs(false);
        cc.setLeftEditable(true);
        cc.setRightEditable(false); cc.setProperty(CompareEditor.CONFIRM SAVE
PROPERTY,new Boolean(false));
        - String leftLabel = PatchMessages.PatchMessages.HunkMergePageInput
WorkspaceCopy;
        - cc.setLeftLabel(leftLabel);
        - String rightLabel = PatchMessages.PatchMessages.HunkMergePageInput
        OrphanedHunk;
        - cc.setRightLabel(rightLabel);
        + if (config != null){
        + cc.setLeftLabel(config.getLeftLabel(config));
        + cc.setLeftImage(config.getLeftImage(config));
        + cc.setRightLabel(config.getRightLabel(config)); +
cc.setRightImage(config.getRightImage(config));
        +}else{
        + String leftLabel = PatchMessages.HunkMergePageInput WorkspaceCopy; +
cc.setLeftLabel(leftLabel);
        + String rightLabel = PatchMessages.HunkMergePageInput OrphanedHunk; +
cc.setRightLabel(rightLabel);
        +} }
}
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class PatcherCompareEditorInput extends CompareEditorInput {
    protected CompareConfiguration config;
    protected  CompareConfiguration cc ;
    public CompareConfiguration getCompareConfiguration() {…}
    private void initLabels() {
    cc = getCompareConfiguration();
    cc.setLeftEditable(false);
    cc.setRightEditable(false);
    extractMethod(config, PatchMessages.PatcherCompareEditorInput
LocalCopy,PatchMessages.PatcherCompareEditorInput AfterPatch);
    }
    protected void extractMethod(Config config, String leftLabel, String rightLabel) {
        if (config != null) {
            cc.setLeftLabel(config.getLeftLabel(config));
            cc.setLeftImage(config.getLeftImage(config));
            cc.setRightLabel(config.getRightLabel(config));
            cc.setRightImage(config.getRightImage(config));
        } else {
            cc.setLeftLabel(leftLabel);
            cc.setRightLabel(rightLabel);
        }
    }
}
public class HunkMergePageInput extends PatcherCompareEditorInput {
    private void initLabels() {
        cc = getCompareConfiguration();
        cc.setCalculateDiffs(false);
        cc.setLeftEditable(true);
        cc.setRightEditable(false);
        cc.setProperty(CompareEditor.CONFIRM_SAVE_PROPERTY,new Boolean(false));
        extractMethod (config, PatchMessages.HunkMergePageInput
WorkspaceCopy,PatchMessages.HunkMergePageInput OrphanedHunk)
        }
}
```

8. Here is an example from org.eclipse.compare.contentmergeviewer. ContentMergeViewer between v20070416 and v20070430.

```
public class ContentMergeViewer {
    protected CompareHandlerService fHandlerService;
    private ICompareInputChangeListener fCompareInputChangeListener;
    public Object getInput(){…}
    protected CompareConfiguration getCompareConfiguration() {…}
    protected void handleDispose(DisposeEvent event) {
    - Utilities.deregisterActions(fHandlerService, fActivations);
    - fHandlerService= null;
    + if (fHandlerService != null)
    + fHandlerService.dispose();
    Object input= getInput();
    if (input instanceof ICompareInput) {
    ICompareContainer container = getCompareConfiguration().getContainer();
    container.removeCompareInputChangeListener((ICompareInput)input,
    fCompareInputChangeListener);
    }
    if (input != null) {
    ICompareInputLabelProvider lp = getCompareConfiguration().getLabelProvider();
    if (lp != null) lp.removeListener(labelChangeListener);
    }
    }
}
public class TextMergeViewer extends ContentMergeViewer {
    static final boolean DEBUG = false;
    private void removeFromDocumentManager(char leg, Object oldInput) {…}
    protected void handleDispose(DisposeEvent event) {
```

```
    - Utilities.deregisterActions(fHandlerService, fActivations);
    - fHandlerService= null;
    + if (fHandlerService != null)
    + fHandlerService.dispose();
    Object input= getInput();
    removeFromDocumentManager(ANCESTOR_CONTRIBUTOR, input);
    removeFromDocumentManager(LEFT_CONTRIBUTOR, input);
    removeFromDocumentManager(RIGHT_CONTRIBUTOR, input);
    if (DEBUG)
    DocumentManager.dump();
    }
 }
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class ContentMergeViewer {
    protected CompareHandlerService fHandlerService;
    private ICompareInputChangeListener fCompareInputChangeListener;
    public Object getInput(){…}
    protected CompareConfiguration getCompareConfiguration() {…}
    protected void handleDispose(DisposeEvent event) {
        extractMethod(fHandlerService);
        Object input = getInput();
        if (input instanceof ICompareInput) {
            ICompareContainer container = getCompareConfiguration().getContainer();
            container.removeCompareInputChangeListener((ICompareInput)
input,fCompareInputChangeListener);
        }
        if (input != null) {
            ICompareInputLabelProvider lp =
getCompareConfiguration().getLabelProvider();
            if (lp != null)
                lp.removeListener(labelChangeListener);
        }
    }
    protected void extractMethod(FileHandleService fHandlerService) {
        if (fHandlerService != null)
        fHandlerService.dispose();
       }
}
public class TextMergeViewer extends ContentMergeViewer {
    static final boolean DEBUG = false;
    private void removeFromDocumentManager(char leg, Object oldInput) {...}
    protected void handleDispose(DisposeEvent event) {
        extractMethod(fHandlerService);
        Object input = getInput();
        removeFromDocumentManager(ANCESTOR_CONTRIBUTOR, input);
        removeFromDocumentManager(LEFT_CONTRIBUTOR, input);
        removeFromDocumentManager(RIGHT_CONTRIBUTOR, input);
        if (DEBUG)
            DocumentManager.dump();
    }
}
```

9. Here is an example from

```
public class CompareEditorInput {
    private ICompareContainer fContainer;
    private boolean fContainerProvided;
    private Splitter fComposite;
    public IActionBars getActionBars() {
   - if (fContainer == null) {
   + IActionBars actionBars = fContainer.getActionBars();
   + if (actionBars == null && !fContainerProvided) {
      return Utilities.findActionBars(fComposite);
     }
   - return fContainer.getActionBars();
   + return actionBars;
  }
}
    public IServiceLocator getServiceLocator() {
   - if (fContainer == null) {
    + IServiceLocator serviceLocator = fContainer.getServiceLocator();
    + if (serviceLocator == null && !fContainerProvided) {
      return Utilities.findSite(fComposite);
     }
    - return fContainer.getServiceLocator();
    + return serviceLocator;
    }
  }
}
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class CompareEditorInput {
    private ICompareContainer fContainer;
    private boolean fContainerProvided;
    private Splitter fComposite;
    public IActionBars getActionBars() {
        ReturnObj obj = extractMethod(fContainer.getServiceLocator(),
Utilities.findSite(fComposite));
        if (obj.type==RETURNTYPE.RETURN)
            return (IActionBars) obj.value;
        return null;
    }
    public IServiceLocator getServiceLocator() {
        ReturnObj obj =
extractMethod(fContainer.getServiceLocator(),Utilities.findSite(fComposite));
        if (obj.type==RETURNTYPE.RETURN)
                return (IServiceLocator) obj.value;
            return null;
    }
    class ReturnObj {
        public RETURNTYPE type;
        public Object value;
        public ReturnObj (RETURNTYPE type, Object value) {
            this.type = type;
            this.value = value;
        }
    }
    enum RETURNTYPE {
        RETURN,FALL;
    }
    private ReturnObj extractMethod (Object object,Object findObj) {
            if (object == null && !fContainerProvided) {
              return new ReturnObj(RETURNTYPE.RETURN, findObj);
            }
            return new ReturnObj(RETURNTYPE.RETURN, object);
    }
}
```

10. Here is an example from

```
public class MergeSourceViewer {
  private HashMap fActions= new HashMap();
  public void textChanged(TextEvent event) {
    Iterator e= fActions.values().iterator();
    while (e.hasNext()) {
      - MergeViewerAction action = (MergeViewerAction)e.next();
      - if (action.isContentDependent())
      -   action.update();
      + Object next = e.next();
      + if (next instanceof MergeViewerAction) {
        + MergeViewerAction action = (MergeViewerAction) next;
        + if (action.isContentDependent())
          + action.update();
        }
    }
  }
  public void selectionChanged(SelectionChangedEvent event) {
    Iterator e= fActions.values().iterator();
    while (e.hasNext()) {
      - MergeViewerAction action = (MergeViewerAction)e.next();
      - if (action.isContentDependent())
    -  action.update();
    + Object next = e.next();
    + if (next instanceof MergeViewerAction) {
     + MergeViewerAction action = (MergeViewerAction) next;
     + if (action.isSelectionDependent())
      + action.update();
     }
   }
  }
 }
```

We extract common code edits with *extractMethod* refactoring. Please specify whether you prefer to apply this code edits with common code extraction refactoring.

```
public class MergeSourceViewer {
 private HashMap fActions= new HashMap();
 public void textChanged(TextEvent event) {
  Iterator e= fActions.values().iterator();
  while (e.hasNext()) {
  extractMethod(action.isContentDependent());
  }
 }
 public void selectionChanged(SelectionChangedEvent event) {
  Iterator e= fActions.values().iterator();
  while (e.hasNext()) {
  extractMethod(action.isSelectionDependent());
  }
 }
 private void extractMethod (boolean flag) {
     Object next = e.next();
     if (next instanceof MergeViewerAction) {
      MergeViewerAction action = (MergeViewerAction) next;
       if (flag)
         action.update();
      }
 }
}
```

11.

class DefaultCommentMapper {

```
Comment[] getLeadingComments(ASTNode node) {
-    if (this.leadingComments != null) {
-        int[] range = (int[])
this.leadingComments.get(node);
+    if (this.leadingPtr >= 0) {
+        int[] range = null;
+        for (int i=0; range==null && i<=this.leadingPtr; i++)
{
+        if (this.leadingNodes[i] == node)
+            range=this.leadingIndexes[i];
+    }
     if (range != null) {
         int length = range[1]-range[0]+1;
         Comment[] leadComments = new Comment[length];

         System.arraycopy(this.comments, range[0],
leadComments, 0, length);
         return  leadComments;
             }
         }
         return null;
     }
Comment[] getTrailingComments(ASTNode node){
-    if (this.trailingComments != null) {
-        int[] range = (int[]) this.trailingComments.get(node);
+    if (this.trailingPtr >= 0) {
+    int[] range = null;
+    for (int i=0; range==null && i<=this.trailingPtr; i++) {
+        if (this.trailingNodes[i] == node)
+            range=this.trailingIndexes[i];
+    }
```

```
        if (range != null) {
            int length = range[1]-range[0]+1;
            Comment[] trailComments = new Comment[length];
            System.arraycopy(this.comments, range[0],
trailComments, 0,length);
            return trailComments;
                 }
            }
            return null;
       }
}
```