Programming Assignment 5

A Simple Text Editor using Java Swing

<Objectives>

To demonstrate an understanding of graphical user interfaces and event-driven programming using Java Swing.

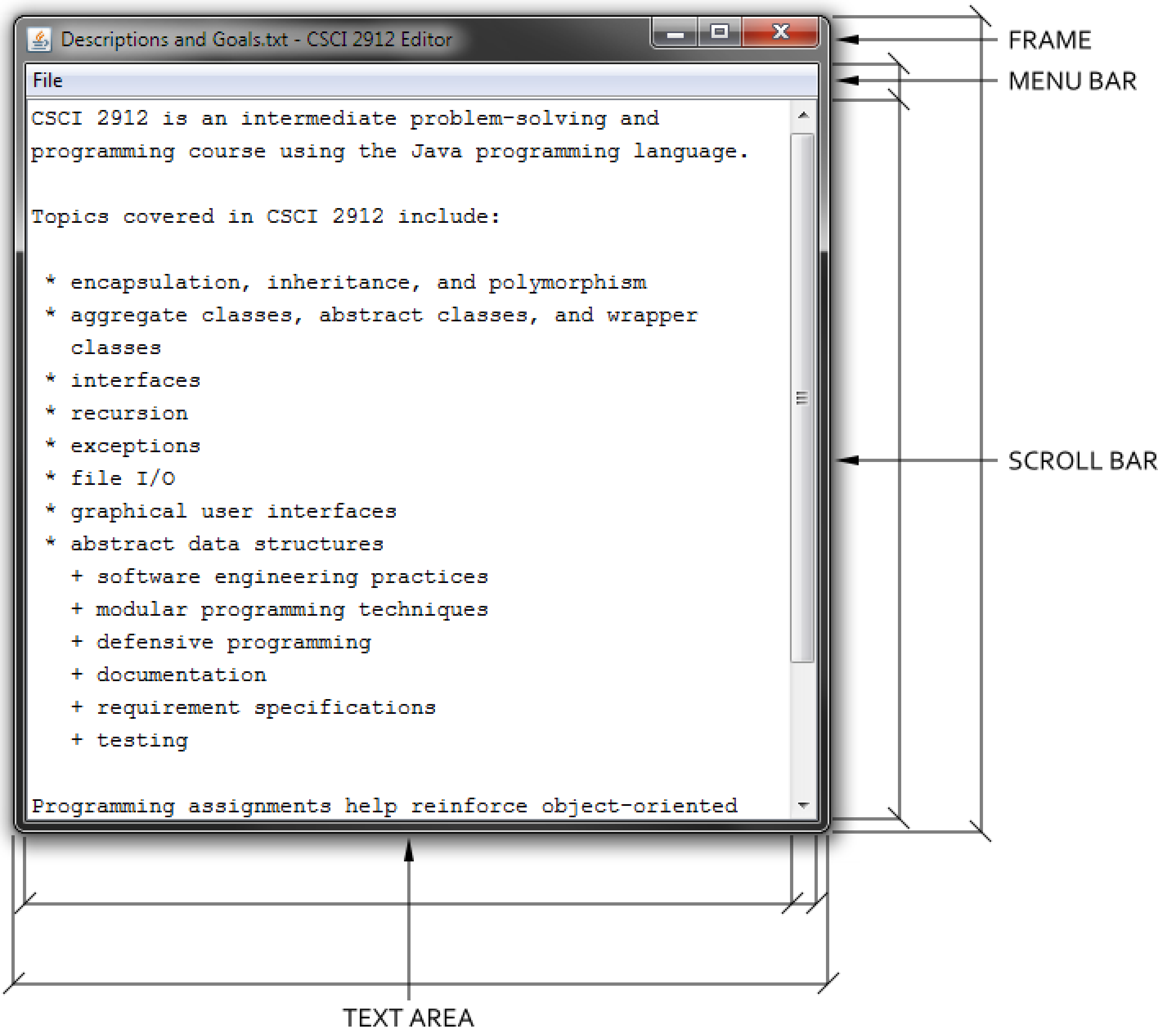
<Requirements>

You will implement a Java Swing application that allows the user to open, edit, and save text files.

When the application starts, it displays a `JFrame` (window) that is centered both horizontally and verticallty on the screen. The application allows the user to move and resize the window using the standard controls provided by the host operating system. Initially, the title of the window is “Untitled - CSCI 2912 Editor”. When the window is closed, the application terminates.

The application sets the content area of the window to a `JScrollPane`. The initial size of the scroll pane is 500x500 pixels. When the window size changes, the scroll pane stretches to fill the entire bounds of the content area.

The application adds to the scroll pane an editable `JTextArea`. The text area displays text with a 14-pt “Monospace” font. The text within the area is left justified and does not wrap automatically. When the size of the scroll pane changes, the text area stretches to fill the entire bounds of the scroll pane. If the size of the text exceeds the size of the scroll pane, horizontal or vertical scroll bars appear that allow the user to scroll through the text area content.



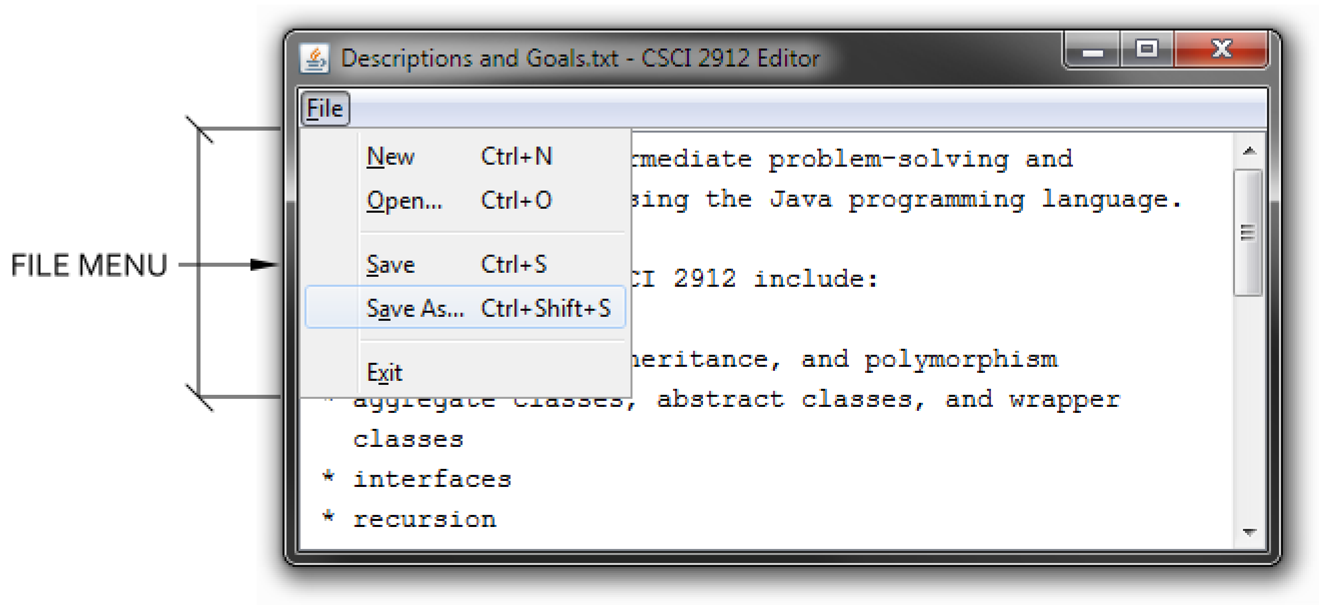
```
CSCI 2912 is an intermediate problem-solving and
programming course using the Java programming language.

Topics covered in CSCI 2912 include:

 * encapsulation, inheritance, and polymorphism
 * aggregate classes, abstract classes, and wrapper
   classes
 * interfaces
 * recursion
 * exceptions
 * file I/O
 * graphical user interfaces
 * abstract data structures
   + software engineering practices
   + modular programming techniques
   + defensive programming
   + documentation
   + requirement specifications
   + testing

Programming assignments help reinforce object-oriented
```

The application displays a `JMenuBar`. The menu bar contains one `JMenu` with the text “File”. The application uses the letter “F” as the mnemonic for the File Menu.

Hint: When hosted on Mac OS X, to cause the Menu Bar to appear at the top of the screen, set the system property “apple.laf.useScreenMenuBar” to “true”. Setting this property on Windows or Linux hosts has no effect.

Hint: When hosted on Mac OS X, Java mnemonics do not appear. When hosted on Windows, Java mnemonics do not appear unless the user press the Alt key.

The File Menu contains five `JMenuItem` instances; ordered from top to bottom, the names of these items are “New”, “Open”, “Save”, “Save As...”, and “Exit”. The application adds a separator between the Open Menu Item and the Save Menu Item, and it adds another separator between the Save As... Menu Item and the Exit Menu Item.



```
rmediate problem-solving and
sing the Java programming language.

CI 2912 include:

heritance, and polymorphism
aggregate classes, abstract classes, and wrapper
  classes
* interfaces
* recursion
```

The application assigns mnemonics to all menu items. The application assigns accelerators to all menu items except the Exit Menu Item. The mnemonics and accelerators are defined in the table below. The application determines the appropriate shortcut key using the `Toolkit.getMenuShortcutKeyMask` method.

Mnemonic	**Text**	**Accelerator**
N	File	[ShortcutKey]+N
O	Open	[ShortcutKey]+O
S	Save	[ShortcutKey]+S
A	Save As...	[ShortcutKey]+Shift+S
X	Exit	(none)

The New Menu Item

When the user selects the New Menu Item, the application clears the text shown in the text area. The application then sets the title of the window to “Untitled - CSCI 2912 Editor”.

The Open Menu Item

When the user selects the Open Menu Item, the application shows an Open File dialog using `JFileChooser`. If the user selects a file from this dialog, the contents of the text area is replaced by the contents of the file. The application then sets the title of the window to “XXX - CSCI 2912 Editor”, where “XXX” is the name of the file not including its directory.

If an I/O error occurs while opening or reading the file, the application does not clear the contents of the text area or change the title of the window but instead displays an error dialog using `JOptionPane.showMessageDialog`. The application sets the title of this dialog to “Error Opening File”, the message type to `JOptionPane.ERROR_MESSAGE`, and the message text itself to the value returned by `Exception.getMessage()`.

The Save Menu Item

When the user selects the Save Menu Item before successfully opening or saving the file since the time the application was started or since the user selected the New Menu Item, then the application proceeds as if the user selected the Save As... Menu Item. Otherwise, the application saves the contents of the text area to the path most recently selected by the user.

If an I/O error occurs while recreating or writing to the file, the application displays an error dialog using `JOptionPane.showMessageDialog`. The application sets the title of this dialog to “Error Saving File”, the message type to `JOptionPane.ERROR_MESSAGE`, and the message text itself to the value returned by `Exception.getMessage()`.

The Save As... Menu Item

When the user selects the Save As... Menu Item, the application shows a Save File dialog using `JFileChooser`. If the user selects a file path from this dialog, the contents of the text area are written to the path selected by the user. The application then sets the title of the window to “XXX - CSCI 2912 Editor”, where “XXX” is the name of the file not including its directory.

If an I/O error occurs while creating or writing the file, the application does not change the title of the window but instead displays an error dialog using `JOptionPane.showMessageDialog`. The application sets the title of this dialog to “Error Saving File”, the message type to `JOptionPane.ERROR_MESSAGE`, and the message text itself to the value returned by `Exception.getMessage()`.

The Exit Menu Item

When the user selects the Exit Menu Item, the application terminates.

<Extra Credit Challenge (10 pt)>

The Save Changes Dialog

You will implement a Save Changes Dialog that asks users if they want to save changes before continuing an operation.

The Save Changes Dialog is a confirmation dialog created using `JoptionPane.showConfirmDialog`. The title of this dialog is “Save Changes?”, the message text of this dialog is “This file has not been saved. Would you like to save it?”, and the option type of this dialog is `JOptionPane.YES_NO_CANCEL_OPTION`.

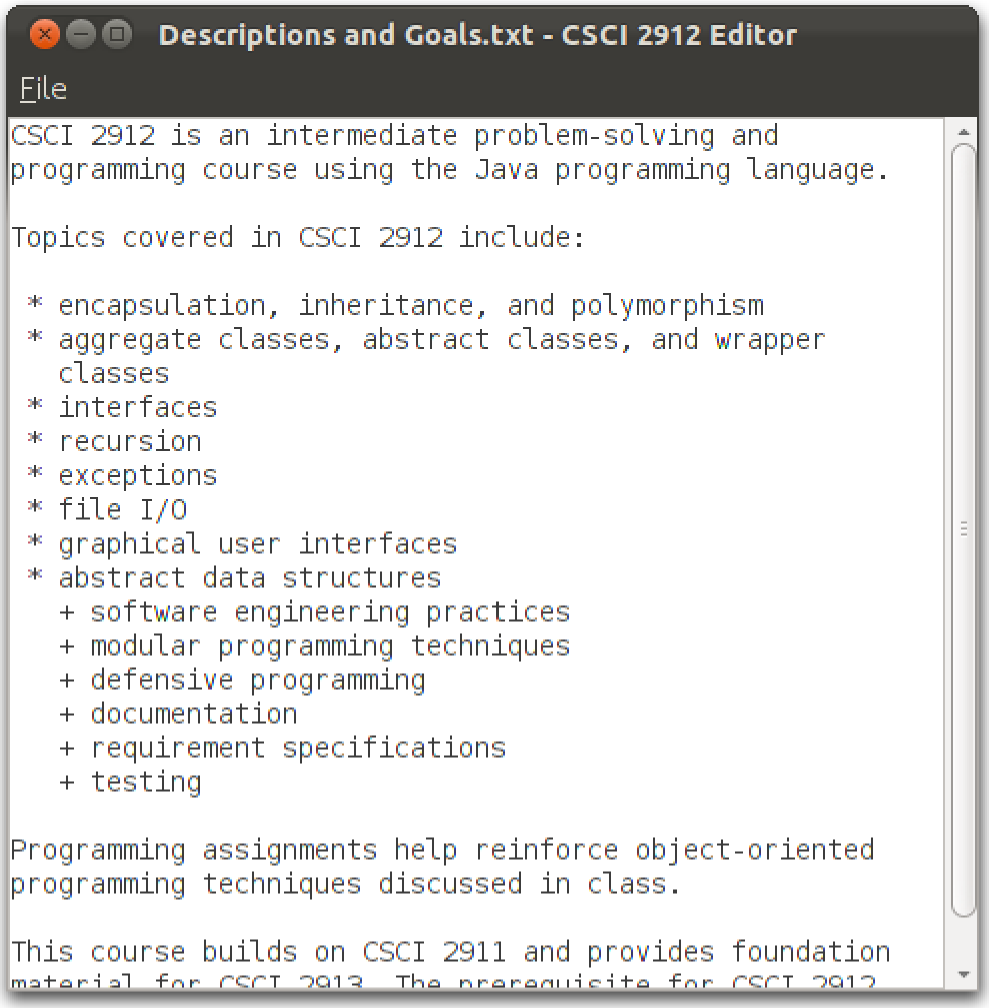
If the user closes the application by selecting the Exit Menu Item or by closing the window, and the contents of the text area have been changed since the file was opened or created, then the application displays the Save Changes Dialog. If the user selects the Cancel Button, the application does not terminate. If the user selects the No Button, the application terminates without saving the changes. If the user selects the Yes Button, the application proceeds as if the user selected the Save Menu Item. The application then terminates if and only if the file is saved successfully.

If the user selects the New Menu Item, and the contents of the text area have been changed since the file was opened or created, then the application displays the Save Changes Dialog. If the user selects the Cancel Button, the application does not start a new file. If the user selects the No Button, the application proceeds normally without saving the changes. If the user selects the Yes Button, the application proceeds as if the user selected the Save Menu Item. The application then starts a new file if and only if the file is saved successfully.

If the user selects the Open Menu Item, and the contents of the text area have been changed since the file was opened or created, then the application displays the Save Changes Dialog. If the user selects the Cancel Button, the application does not open a new file. If the user selects the No Button, the application proceeds normally without saving the changes. If the user selects the Yes Button, the application proceeds as if the user selected the Save Menu Item. The application then opens a new file if and only if the file is saved successfully.

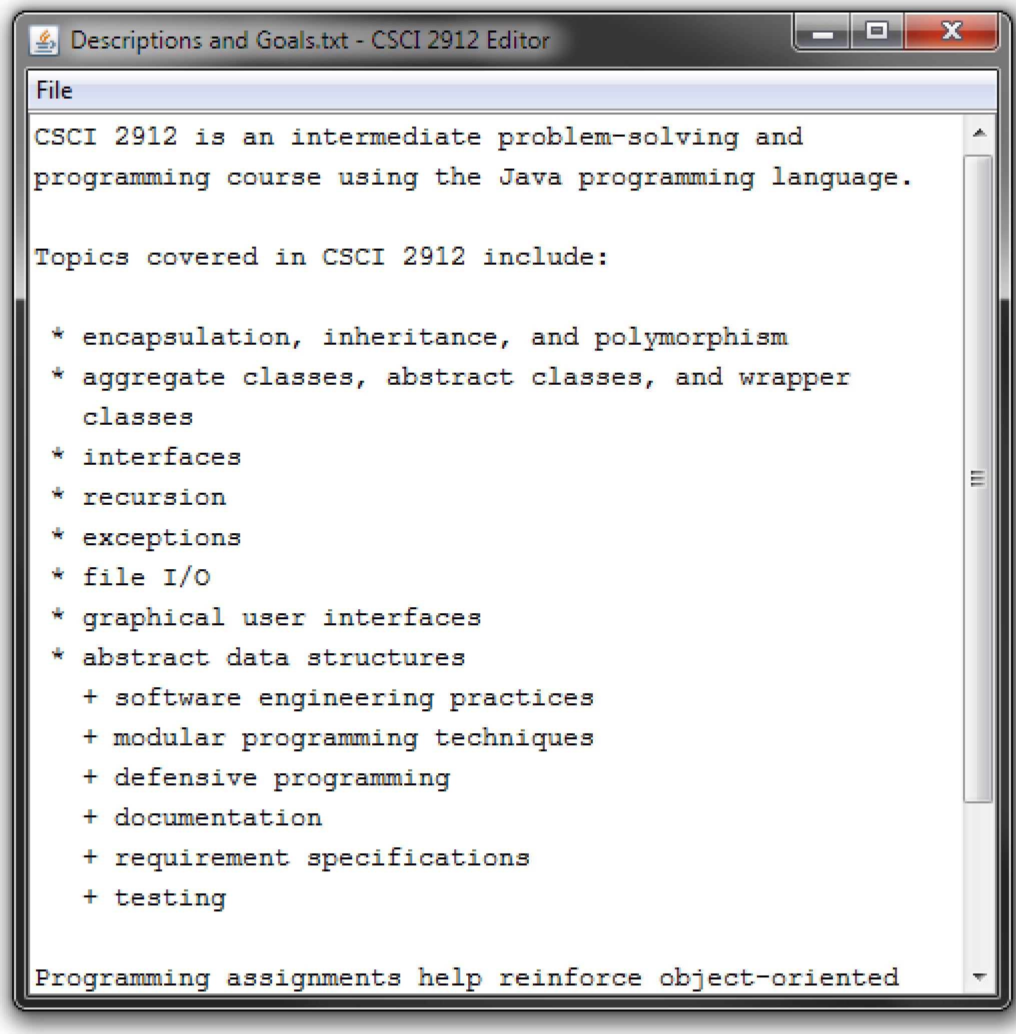
<Screenshots>

Linux



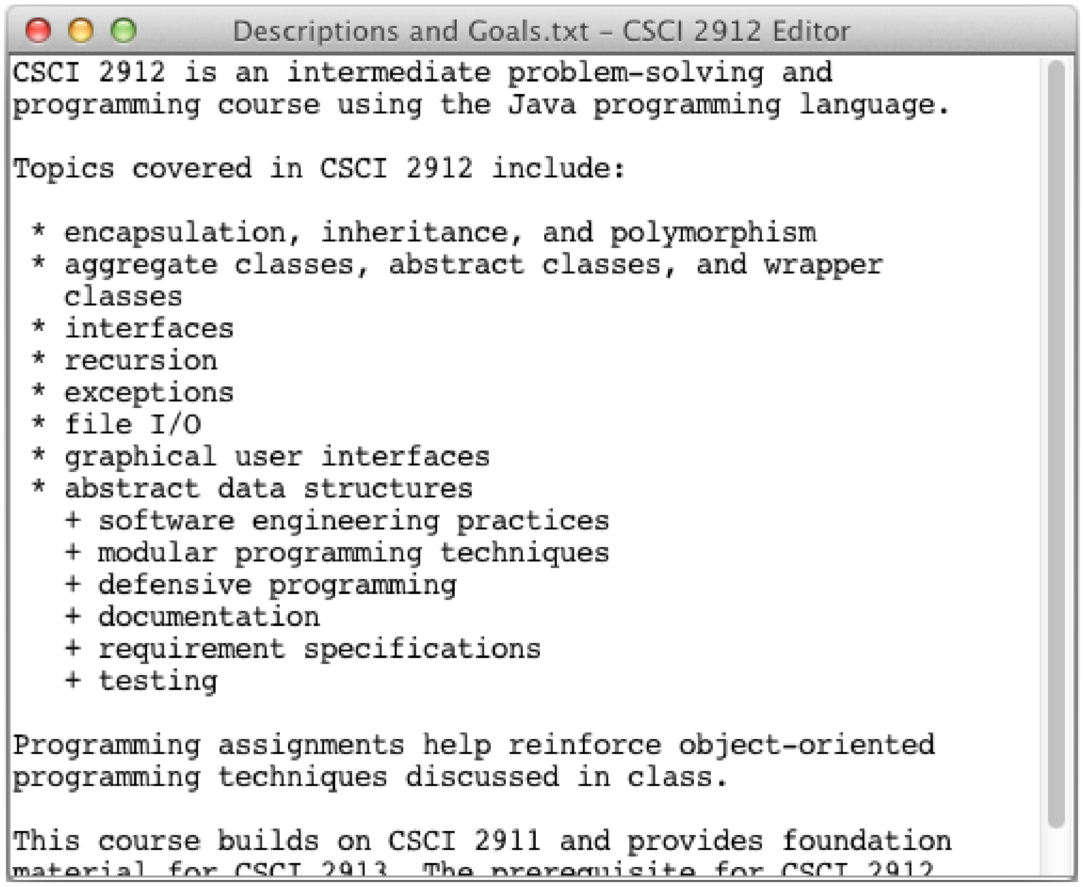
```
CSCI 2912 is an intermediate problem-solving and
programming course using the Java programming language.

Topics covered in CSCI 2912 include:

 * encapsulation, inheritance, and polymorphism
 * aggregate classes, abstract classes, and wrapper
   classes
 * interfaces
 * recursion
 * exceptions
 * file I/O
 * graphical user interfaces
 * abstract data structures
   + software engineering practices
   + modular programming techniques
   + defensive programming
   + documentation
   + requirement specifications
   + testing

Programming assignments help reinforce object-oriented
programming techniques discussed in class.

This course builds on CSCI 2911 and provides foundation
material for CSCI 2913. The prerequisite for CSCI 2912
```

Windows



```
CSCI 2912 is an intermediate problem-solving and
programming course using the Java programming language.

Topics covered in CSCI 2912 include:

 * encapsulation, inheritance, and polymorphism
 * aggregate classes, abstract classes, and wrapper
   classes
 * interfaces
 * recursion
 * exceptions
 * file I/O
 * graphical user interfaces
 * abstract data structures
   + software engineering practices
   + modular programming techniques
   + defensive programming
   + documentation
   + requirement specifications
   + testing

Programming assignments help reinforce object-oriented
```

Mac OS



Descriptions and Goals.txt – CSCI 2912 Editor

```
CSCI 2912 is an intermediate problem-solving and
programming course using the Java programming language.

Topics covered in CSCI 2912 include:

 * encapsulation, inheritance, and polymorphism
 * aggregate classes, abstract classes, and wrapper
   classes
 * interfaces
 * recursion
 * exceptions
 * file I/O
 * graphical user interfaces
 * abstract data structures
   + software engineering practices
   + modular programming techniques
   + defensive programming
   + documentation
   + requirement specifications
   + testing

Programming assignments help reinforce object-oriented
programming techniques discussed in class.

This course builds on CSCI 2911 and provides foundation
material for CSCI 2913. The prerequisite for CSCI 2912
```

<Resources>

- Relevant API (java.lang)
    - Runnable
    - System
- Relevant API (java.io)
    - ByteArrayInputStream
    - File
    - FileInputStream
    - FileWriter
    - IOException
- Relevant API (java.awt)
    - ActionEvent
    - ActionListener
    - Dimension
    - Font
    - KeyEvent
    - Toolkit
- Relevant API (javax.swing)
    - KeyStroke
    - JFileChooser
    - JFrame
    - JMenu
    - JMenuBar
    - JMenuItem
    - JOptionPane
    - JScrollPane
    - JTextArea
    - SwingUtilities
    - UIManager