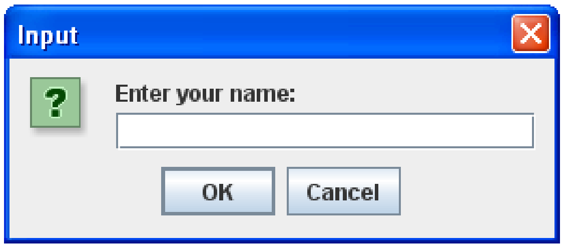
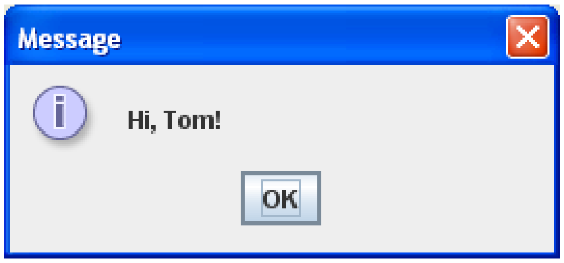
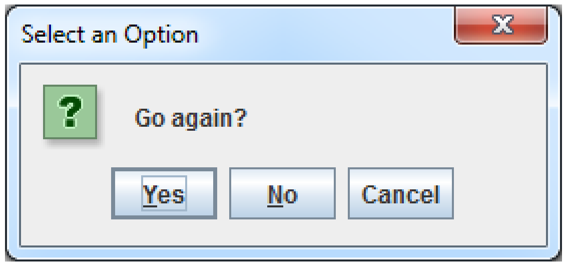
javax.swing.JOptionPane

The Java API class `javax.swing.JOptionPane` has facilities for creating a ***dialog box*** that can appear on the computer's desktop to request input from or display messages to the user. Here are three easy dialog boxes provided by the class.

Three Types of JOptionPane Dialog Boxes

Dialog Box	**Method**	**Description/Example**
Input Dialog	`showInputDialog`	Prompts user for input  Enter your name:
Message Dialog	`showMessageDialog`	Displays a message to the user  Hi, Tom!
Confirm Dialog	`showConfirmDialog`	Asks the user for his or her consent  Go again?

Message Dialogs

JOptionPane Methods to Display a Message Dialog

`static void showMessageDialog( Component win, String message,`
`      String title, int messageType, Icon icon )`
`// Displays a message dialog with the specified message,`
`// title, message type and icon.`

`static void showMessageDialog( Component win, String message,`
`      String title, int messageType )`
`// Displays a message dialog with a given message, title and`
`// message type, which indicates which icon to display.`

`static void showMessageDialog( Component win, String message )`
`// Displays a message dialog titled "Message" and showing the`
`// default icon.`

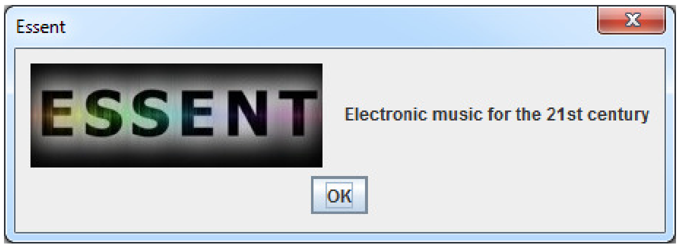
Parameters for showMessageDialog

Parameter	Description
`Component win`	A reference to your application's desktop window; if your application doesn't have one, pass it the `null` pointer.
`String message`	The message you want displayed to the user.
`String title`	The title that is to be displayed in the title bar of the dialog box.
`int messageType`	An integer code indicating the type of message to be displayed. This is used primarily to pick from among a preselected set of icons.
`Icon icon`	An icon to display within the dialog box.

The dialog box is ***modal***, meaning that the execution of the Java program is blocked until the user's interaction with the box is completed, which happens when the user clicks the OK button or the Close button (❎).

Example

The Java application below displays the message dialog shown at right.



Electronic music for the 21st century

The following table shows the five arguments passed to the method and why.

Arguments Passed to showMessageDialog and Why

Argument	Why
`null`	The application has no desktop window.
`msg`	Contains the message.
`ttl`	Contains the title.
`0`	Doesn't matter since I'm passing the icon I want to use.
`icon`	The icon I want to display within the dialog box.

```
import static javax.swing.JOptionPane.*;

public class MyApp
{
   public static void main( String [] args )
   {
      Icon icon = new ImageIcon( "essent.jpg" );
      String ttl = "Essent";
      String msg = "Electronic music for the 21st century";
      showMessageDialog( null, msg, ttl, 0, icon );
   }
}
```

If you don't want to go to the trouble of creating your own icon, the second overloaded `showMessageDialog` method allows you to choose an icon from among a preselected set. You do this by passing an integer code as the fourth argument.

It is not considered good programming sportsmanship to require fellow programmers to remember the meanings of specific integer codes. Instead, seasoned programmers provide ***ease-of-use constants***, which are predefined constant identifiers that his or her fellow programmers can pass as argument values.

In the Java API, ease-of-use constants are usually defined as static fields within the class. A list of static fields within the `javax.swing.JOptionPane` that are valid for the message type parameter is shown on the next page.

To use these constants, simply pass them as the fourth argument to the `showMessageDialog` method.

Example

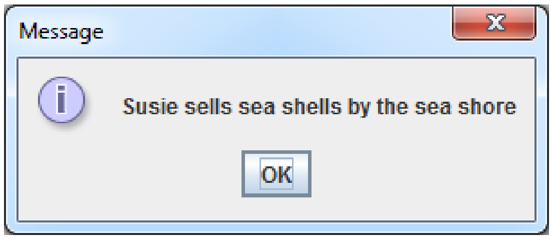
These Java statements display the first message dialog shown in the table on the next page.

```
import static javax.swing.JOptionPane.*;
. . .
showMessageDialog( null, "Message", "Title", ERROR_MESSAGE );
```

The third overloaded `showMessageDialog` method is very short and uses a default title and icon.

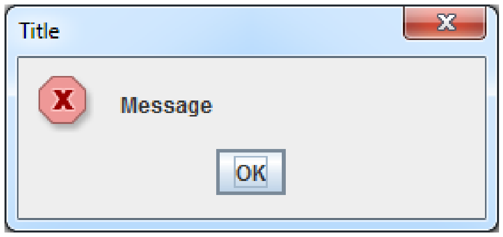
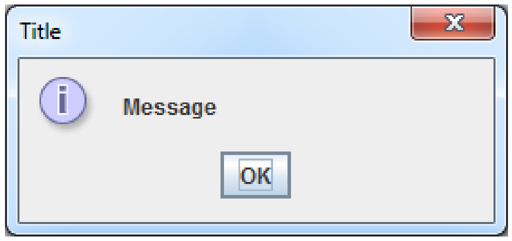
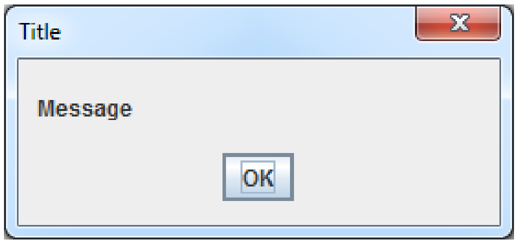
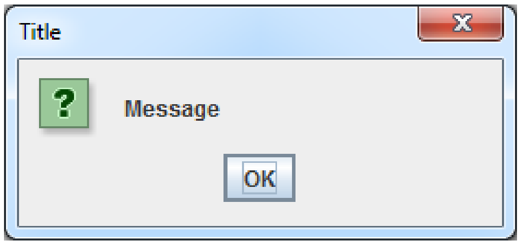
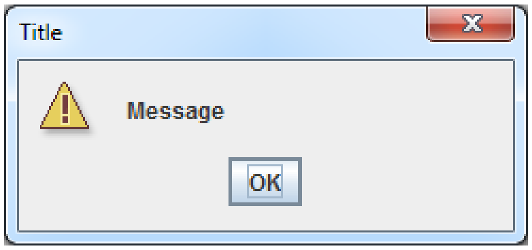
Example

The Java statements below display the output dialog shown at right.



Susie sells sea shells by the sea shore

```
String msg = "Susie sells sea shells by the sea shore";
javax.swing.JOptionPane.showMessageDialog( null, msg );
```

JOptionPane Ease-Of-Use Constants that Specify Message Types	
Constant Identifier	**Meaning /Example**
`static int ERROR_MESSAGE`	Displays an error icon 
`static int INFORMATION_MESSAGE`	Displays an information icon 
`static int PLAIN_MESSAGE`	Doesn't display any icon 
`static int QUESTION_MESSAGE`	Displays a question mark icon 
`static int WARNING_MESSAGE`	Displays a warning message icon 

Input Dialogs

JOptionPane Methods to Display an Input Dialog

`static String showInputDialog( Component win, String prompt, String title, int messageType )`
`// Displays a dialog requesting input from the user`
`// with the given prompt message, title and message type.`

`static String showInputDialog( Component win, String prompt, String defaultInput )`
`// Displays a dialog requesting input from`
`// the user with the given prompt and default input.`

`static String showInputDialog( Component win, String prompt )`
`// Displays a dialog requesting input from`
`// the user with the given prompt.`

`static String showInputDialog( String prompt, String defaultInput )`
`// Displays a dialog requesting input from`
`// the user with the given prompt and default input.`

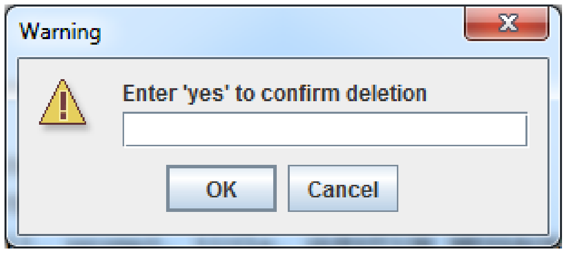
`static String showInputDialog( String prompt )`
`// Displays a dialog requesting input from`
`// the user with the given prompt.`

Parameters for showInputDialog

Parameter	Description
`Component win`	If this argument is present, it must be a reference to a GUI component within your application. The input dialog is displayed centered over the component. If this argument is absent, the input dialog is centered over the desktop.
`String prompt`	The prompt message you want displayed to the user.
`String title`	The title that is to be displayed in the title bar of the dialog box.
`int messageType`	An integer code indicating the type of message as explained for `showMessageDialog`.
`String defaultInput`	A value to appear in the dialog as a default input value.

Example

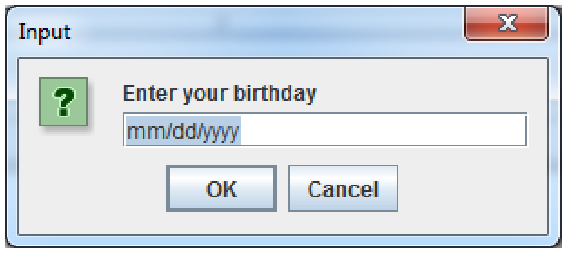
The Java application below displays the message dialog shown at right.


Enter 'yes' to confirm deletion

```
import static javax.swing.JOptionPane.*;

public class MyApp
{
   public static void main( String [] args )
   {
      String prompt = "Enter 'yes' to confirm deletion";
      String title = "Warning";
      String input = showInputDialog
                  ( null, prompt, title, WARNING_MESSAGE );
   }
}
```

Example

These statements display this input dialog.

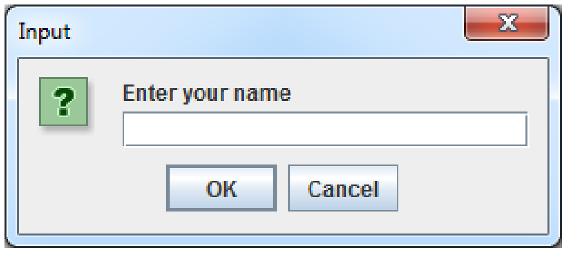

Enter your birthday

mm/dd/yyyy

```
String prompt = "Enter your birthday";
String input = showInputDialog( prompt, "mm/dd/yyyy"  );
```

Example

This statement displays this input dialog.


Enter your name

```
String input = showInputDialog( "Enter your name" );
```

An input dialog is ***modal***, blocking the Java program until the user's interaction with the box is completed, which happens as the following table explains:

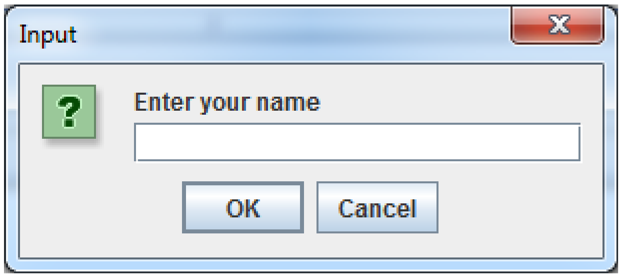
The Effect of User Actions on an Input Dialog Box

User Action	Effect
Clicks the OK button with the mouse	A **`String`** object is returned containing whatever string is in the dialog's text field
Presses the Enter key on the keyboard	
Clicks the Cancel button	A null pointer is returned
Clicks the Close button (☒)	

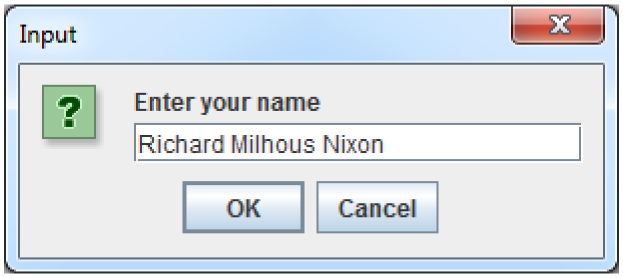
Example

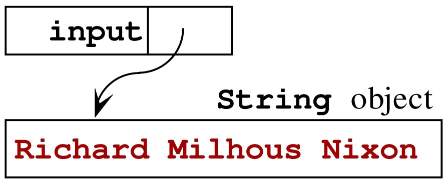
This statement: **`String input = showInputDialog( "Enter your name" );`**

Displays this dialog:

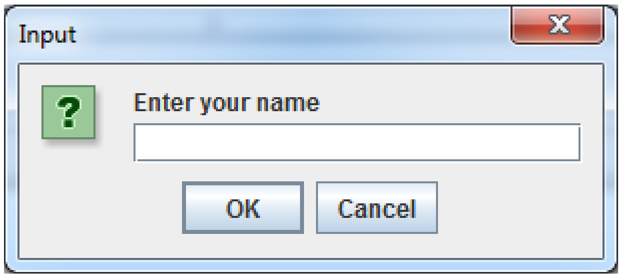


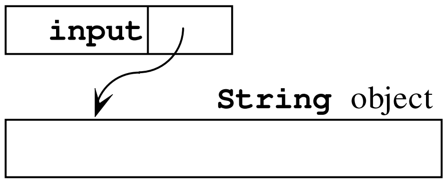
If the user types within the text field and clicks OK or presses Enter, any contents of the text field is placed into a **`String`** object and its reference placed into variable **`input`**.



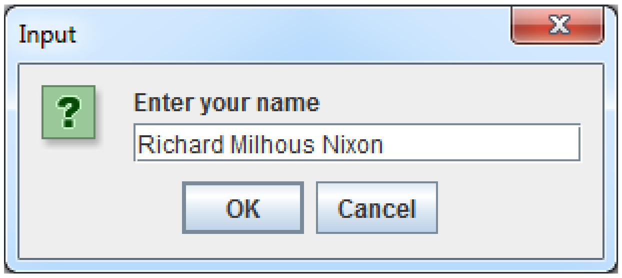


If the user clicks OK or presses Enter with nothing in the text box, a **String** object containing the null string is returned.

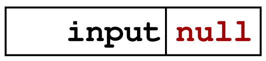




If the user clicks Cancel or Close (no matter what's in the text field), the null pointer is returned.



Richard Milhous Nixon



Confirm Dialogs

JOptionPane Methods to Display a Confirm Dialog

`static int showConfirmDialog( Component win, String message, String title, int optionType, int messageType, Icon icon )`
`// Displays a confirm dialog with the given message, title,`
`// list of options (specified by the option type) and icon.`

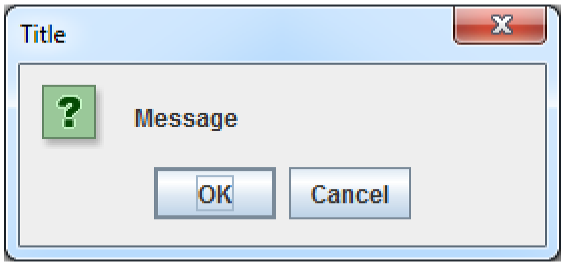
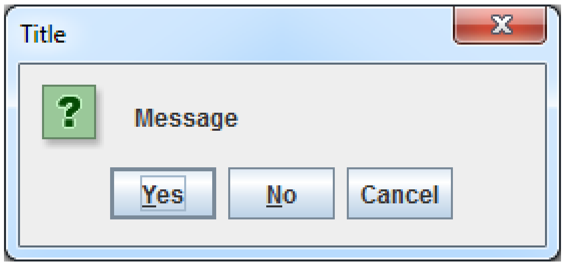
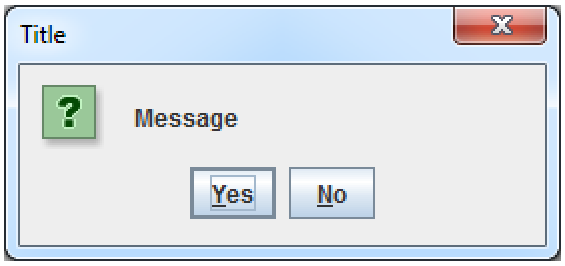
`static int showConfirmDialog( Component win, String message, String title, int optionType, int messageType )`
`// Displays a confirm dialog with a given message, title,`
`// list of options (specified by the option type) and`
`// message type, which indicates which icon to display.`

`static int showConfirmDialog( Component win, String message, String title, int optionType )`
`// Displays a confirm dialog with the given message, title`
`// and options as specified by the given option type.`

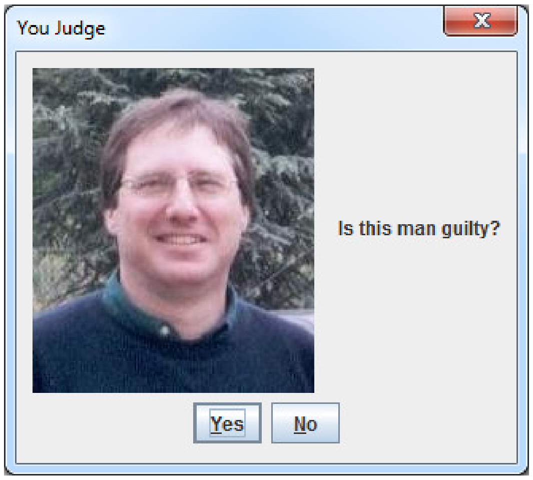
`static int showConfirmDialog( Component win, String message )`
`// Displays a confirm dialog containing the given message and`
`// the options Yes, No and Cancel.`

Parameters for showConfirmDialog

Parameter	Description
`Component win`	A reference to your application's desktop window; if your application doesn't have one, pass it the `null` pointer.
`String message`	The prompt message you want displayed to the user.
`String title`	The title that is to be displayed in the title bar of the dialog box. If omitted the title "Select an Option" is displayed.
`int optionType`	An integer code indicating the desired array of options to be presented to the user on the dialog.
`int messageType`	An integer code indicating the type of message as explained for `showMessageDialog`. If omitted, the QUESTION_MESSAGE icon is displayed.
`Icon icon`	An icon to display within the dialog box as explained for `showMessageDialog`.

JOptionPane Ease-Of-Use Constants that Specify Option Types	
Constant Identifier	**Example**
`static int OK_CANCEL_OPTION`	 Title Message OK Cancel
`static int YES_NO_CANCEL_OPTION`	 Title Message Yes No Cancel
`static int YES_NO_OPTION`	 Title Message Yes No

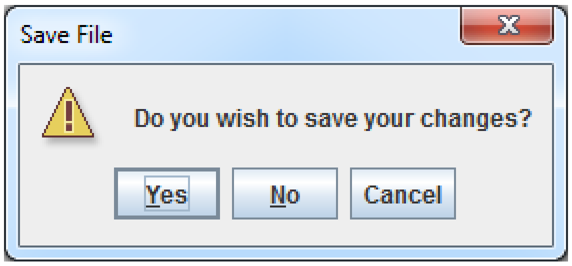
Example
The Java application below displays the message dialog shown at right.



```
import javax.swing.*;
import static javax.swing.JOptionPane.*;

public class MyApp
{
   public static void main( String [] args )
   {
      Icon icon = new ImageIcon( "mertz.jpg" );
      String ttl = "You Judge";
      String msg = "Is this man guilty?";
      int ans = showConfirmDialog
         ( null, msg, ttl, YES_NO_OPTION, 0, icon );
   }
}
```

Example

These statements display this confirm dialog.



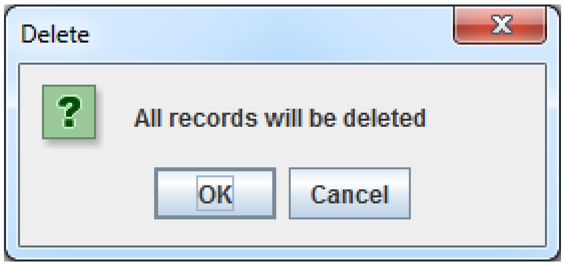
Save File

Do you wish to save your changes?

```
String ttl = "Save File";
String msg = "Do you wish to save your changes?";
int ans = showConfirmDialog
   ( null, msg, ttl, YES_NO_CANCEL_OPTION, WARNING_MESSAGE );
```

Example

This statement displays this confirm dialog.



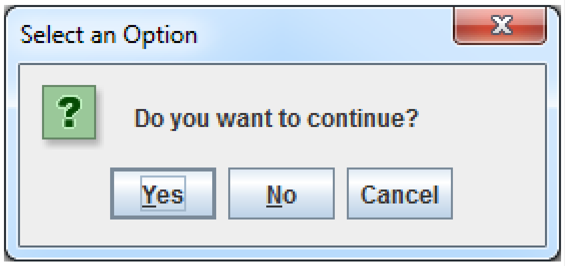
Delete

All records will be deleted

```
String ttl = "Delete";
String msg = "All records will be deleted";
int ans = showConfirmDialog
   ( null, msg, ttl, OK_CANCEL_OPTION );
```

Example

These statements displays this confirm dialog.



Do you want to continue?

```
int ans;
ans = showConfirmDialog( null, "Do you want to continue?" );
```

A confirm dialog box is ***modal***, blocking the Java program until the user clicks one of the option buttons or the Close (☒) button. The **`showConfirmDialog`** method returns an integer code indicating which button the user clicked, which your program can check using an **`if-else`** or **`switch`** statement.

JOptionPane Ease-Of-Use Constants that Specify showConfirmDialog Return Values

Constant Identifier	Meaning
`static int CANCEL_OPTION`	User clicked Cancel button
`static int CLOSE_OPTION`	User clicked Close button
`static int NO_OPTION`	User clicked No button
`static int YES_OPTION`	User clicked Yes button

Example

```
int ans;
ans = showConfirmDialog( null, "Do you want to continue?" );
switch ( ans ) {
   case CANCEL_OPTION:
      . . .
   case CLOSE_OPTION:
      . . .
   case NO_OPTION:
      . . .
   case YES_OPTION:
      . . .
}
```

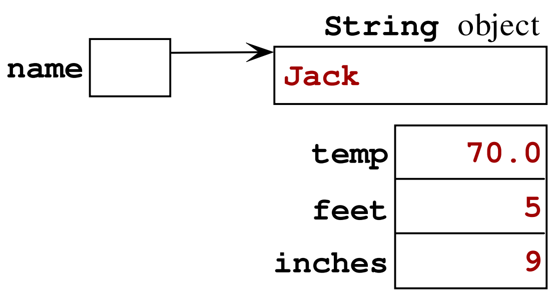
Exercises

Enter the application given below into jGRASP, save it to a file and compile it. Do the exercises that follow.

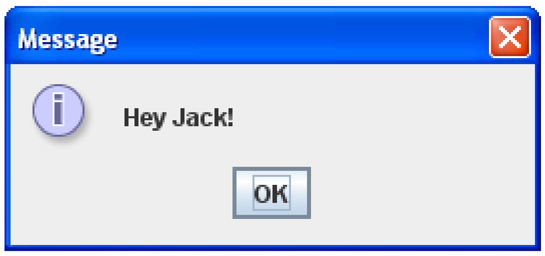
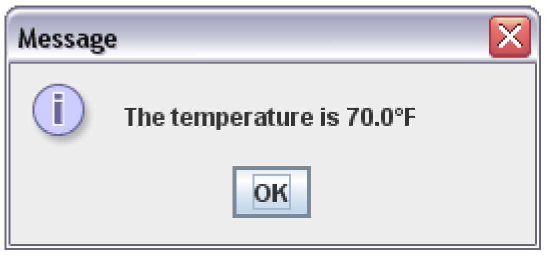
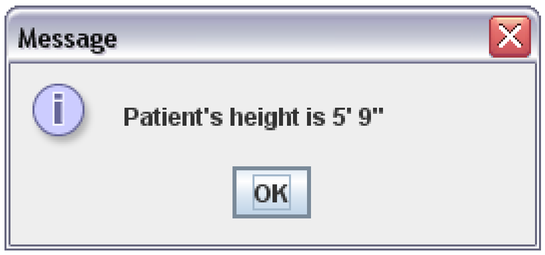
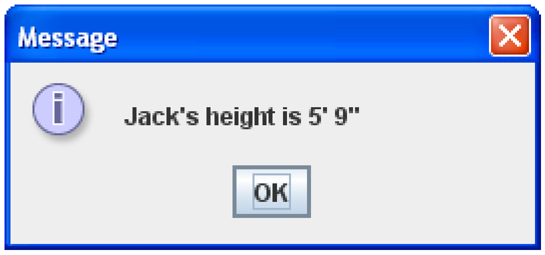
```
import static javax.swing.JOptionPane.*;

public class MyApp
{
   public static void main( String [] args )
   {
      String prompt, name, out;
      prompt = "What's your name?";
      name = showInputDialog( prompt );
      out = "Welcome to Java, " + name + "!";
      showMessageDialog( null, out  );
   }
}
```

1.	Run the program; enter **Jack** in the text field; click the OK button. Observe the output.
2.	Run the program; enter **Jack** in the text field; press the Enter key. Observe the output.
3.	Run the program; click the OK button without entering anything in the text field. Observe the output.
4.	Run the program; press the Enter key without entering anything in the text field. Observe the output.
5.	Run the program; click the Cancel button. Observe the run-time error message. Explain the error.
6.	Run the program; click the Close button. Observe the run-time error message. Explain the error.

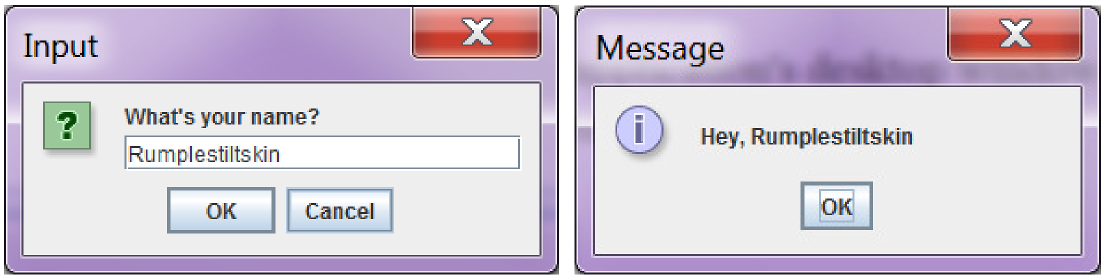
Assume that variables **name**, **temp**, **feet** and **inches** have these values:



For each of the following output dialogs, write the call to **showMessageDialog** to display it. Construct the output string using the variables given above.

7.	 Hey Jack!
8.	 The temperature is 70.0°F
9.	 Patient's height is 5' 9"
10.	 Jack's height is 5' 9"

For each of the following, write the Java statements to read the string from a **`JOptionPane`** input dialog. Declare whatever variables are necessary.	
11.	An address such as *1600 Pennsylvania Avenue NW*.
12.	A city such as *Washington, D.C.*
13.	A state such as *New Jersey*.

14.	Use **`JOptionPane`** dialog boxes to read the user's name and print a greeting. For example:  What's your name? Rumplestiltskin Hey, Rumplestiltskin