```java
public class Song
{
    private int duration;
    private String artist;
    private String title;

    public Song(String artist, String title, int duration)
    {
        this.artist = artist;
        this.title = title;
        this.duration = duration;
    }

    public int getDuration()
    {
        return duration;
    }

    public String getArtist()
    {
        return artist;
    }

    public String getTitle()
    {
        return title;
    }

    public String toString()
    {
        return artist + " - " + title + " (" + duration + "s)";
    }
}
```

```java
import java.util.ArrayList;
public class SongList
{
    private ArrayList<Song> alSongs = new ArrayList<>();

    public int calculateDuration()
    {
        int total = 0;
        for (int i = 0; i < alSongs.size(); i++)
            total += alSongs.get(i).getDuration();
        return total;
    }

    public boolean add(Song song)
    {
        boolean found = false;
        int i = 0;
        while (!found && i < alSongs.size())
        {
            // autre version (meilleure, plus exacte):
            //    song.getArtist().equals(s.getArtist()) && song.getTitle().equals(s.getTitle()) && song.getDuration() == s.getDuration())

            // plus rapide, mais marche ici
            if (song.toString().equals(alSongs.get(i).toString()))
                found = true;
            else
                i++;
        }
        if (found)
        {
            // song déjà dans la liste
            return false;
        }
        else
        {
            alSongs.add(song);
            return true;
        }
    }

    public Song get(int index)
    {
        return alSongs.get(index);
    }

    public Object[] toArray()
    {
        return alSongs.toArray();
    }

    public Song findLongestSong()
    {
        Song longest = null;
        Song current;
        if (alSongs.size() > 0)
        {
            longest = alSongs.get(0);
            for (int i = 1; i < alSongs.size(); i++)
            {
                current = alSongs.get(i);
                if (current.getDuration() > longest.getDuration())
                    longest = current;
            }
        }
        return longest;
    }

    public int findIndexOfLongestSong()
    {
        int longestIndex = -1;
        if (alSongs.size() > 0)
        {
            longestIndex = 0;
            for (int i = 1; i < alSongs.size(); i++)
            {
                if (alSongs.get(i).getDuration() > alSongs.get(longestIndex).getDuration())
                    longestIndex = i;
            }
        }
        return longestIndex;
    }





        // suite, page suivante
```

```java
    public void sort()
    {
        int posMin;
        String min, current;
        int n = alSongs.size();
        for (int i = 0; i <= n - 2; i++)
        {
            posMin = i;
            min = alSongs.get(i).getArtist() + alSongs.get(i).getTitle();
            for (int j = i + 1; j <= n - 1; j++)
            {
                current = alSongs.get(j).getArtist() + alSongs.get(j).getTitle();
                if (current.compareTo(min) < 0)
                {
                    posMin = j;
                    min = current;
                }
            }
            if (posMin != i)
            {
                Song temp = alSongs.get(i);
                alSongs.set(i, alSongs.get(posMin));
                alSongs.set(posMin, temp);
            }
        }
    }

    public LimitedPlayList compilePlayList(String artist, int capacity)
    {
        Song s;
        LimitedPlayList playList = new LimitedPlayList(capacity);
        for (int i = 0; i < alSongs.size(); i++)
        {
            s = alSongs.get(i);
            if (artist.equals(s.getArtist()))
                playList.add(s);
        }
        return playList;
    }
}
```

```java
public class LimitedPlayList extends SongList
{
    private int capacity;

    public LimitedPlayList()
    {
        // capacité par defaut
        capacity = 74 * 60;
    }

    public LimitedPlayList(int capacity)
    {
        this.capacity = capacity;
    }

    public LimitedPlayList(double capacity)
    {
        // convertir minutes -> secondes
        this.capacity = (int) (capacity * 60);
    }

    public int getCapacity()
    {
        return capacity;
    }

    public int getFreeCapacity()
    {
        return capacity - calculateDuration();
    }

    public boolean add(Song song)
    {
        if (calculateDuration() + song.getDuration() <= capacity)
            return super.add(song);
        else
            return false;
    }
}
```

```java
import javax.swing.JOptionPane;
public class MainFrame extends javax.swing.JFrame
{
    private SongList songList = null;
    private LimitedPlayList playList = null;

    public MainFrame()
    {
        initComponents();

        songList = new SongList();
        playList = new LimitedPlayList(74.0);


        // Ajouter des données de test... plus simple que de le faire à la main
        // TEST - begin
        songList.add(new Song("AC/DC", "Live Wire", 305));
        songList.add(new Song("Lynyrd Skynyrd", "Sweet Home Alabama", 285));
        songList.add(new Song("Queen", "Play the Game", 213));
        songList.add(new Song("AC/DC", "High Voltage", 258));
        songList.add(new Song("Queen", "Somebody to Love", 296));
        songList.add(new Song("AC/DC", "Highway to Hell", 208));
        songList.add(new Song("Queen", "Another One Bites the Dust", 216));
        songList.add(new Song("AC/DC", "T.N.T.", 238));
        songList.add(new Song("Queen", "We Will Rock You", 121));
        songList.add(new Song("AC/DC", "Thundersruck", 292));
        songList.add(new Song("AC/DC", "It's a Long Way to the Top", 316));
        songList.add(new Song("Iron Maiden", "Run to the Hills", 230));
        songList.add(new Song("Queen", "Bicycle Race", 181));
        songList.add(new Song("Deep Purple", "Smoke on the Water", 343));
        songList.add(new Song("Queen", "Bohemian Rhapsody", 357));
        songList.add(new Song("Metallica", "Nothing Else Matters", 388));
        songList.add(new Song("Queen", "Crazy Little Thing Called Love", 162));
        songList.add(new Song("AC/DC", "Rock or Bust", 183));
        songList.add(new Song("Gary Moore", "Parisienne Walkways", 407));
        songList.add(new Song("Queen", "We Are the Champions", 180));
        songList.add(new Song("Santana", "Black Magic Woman", 197));
        songList.add(new Song("Iron Maiden", "The Number of the Beast", 265));
        songList.add(new Song("Gary Moore", "Still Got the Blues (For You)", 250));
        // TEST - end


        updateView();
    }

    public void updateView()
    {
        allSongsList.setListData(songList.toArray());
        playlistList.setListData(playList.toArray());

        durationLabel.setText(songList.calculateDuration() + "s");
        playlistDurationLabel.setText(playList.calculateDuration() + "s");
        playlistFreeLabel.setText(playList.getFreeCapacity() + "s");
        playlistTotalLabel.setText(playList.getCapacity() + "s");
    }
// Skipped: ... initComponents { ... }
    private void addButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_addButtonActionPerformed
    {//GEN-HEADEREND:event_addButtonActionPerformed
        Song song = new Song(artistTextField.getText(), titleTextField.getText(), Integer.valueOf(durationTextField.getText()));
        if (!songList.add(song))
            JOptionPane.showMessageDialog(rootPane, "The selected song is already in the song-list!");

        updateView();
    }//GEN-LAST:event_addButtonActionPerformed

    private void findButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_findButtonActionPerformed
    {//GEN-HEADEREND:event_findButtonActionPerformed
        int i = songList.findIndexOfLongestSong();
        if (i >= 0)
            allSongsList.setSelectedIndex(i);
    }//GEN-LAST:event_findButtonActionPerformed

    private void sortButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_sortButtonActionPerformed
    {//GEN-HEADEREND:event_sortButtonActionPerformed
        songList.sort();
        updateView();
    }//GEN-LAST:event_sortButtonActionPerformed


        // suite, page suivante
```

```java
    private void addToPlayListButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_addToPlayListButtonActionPerformed
    {//GEN-HEADEREND:event_addToPlayListButtonActionPerformed
        if (allSongsList.getSelectedIndex() < 0)
            return;

        Song song = songList.get(allSongsList.getSelectedIndex());
        if (!playList.add(song))
            JOptionPane.showMessageDialog(rootPane, "The selected song is already in the play-list or there is not enough space left!");

        updateView();
    }//GEN-LAST:event_addToPlayListButtonActionPerformed

    private void compileButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_compileButtonActionPerformed
    {//GEN-HEADEREND:event_compileButtonActionPerformed
        playList = songList.compilePlayList(artistTextField.getText(), 74 * 60);
        updateView();
    }//GEN-LAST:event_compileButtonActionPerformed

    private void allSongsListValueChanged(javax.swing.event.ListSelectionEvent evt)//GEN-FIRST:event_allSongsListValueChanged
    {//GEN-HEADEREND:event_allSongsListValueChanged
        // Automatically show selected song in fields
        if (allSongsList.getSelectedIndex() >= 0)
        {
            Song song = songList.get(allSongsList.getSelectedIndex());
            titleTextField.setText(song.getTitle());
            artistTextField.setText(song.getArtist());
            durationTextField.setText(String.valueOf(song.getDuration()));
        }
        else
        {
            // ... or clear the fields
            titleTextField.setText("");
            artistTextField.setText("");
            durationTextField.setText("");
        }
    }//GEN-LAST:event_allSongsListValueChanged
// Skipped: ... Look & Feel
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton addButton;
    private javax.swing.JButton addToPlayListButton;
    private javax.swing.JList allSongsList;
    private javax.swing.JTextField artistTextField;
    private javax.swing.JButton compileButton;
    private javax.swing.JLabel durationLabel;
    private javax.swing.JTextField durationTextField;
    private javax.swing.JButton findButton;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JLabel playlistDurationLabel;
    private javax.swing.JLabel playlistFreeLabel;
    private javax.swing.JList playlistList;
    private javax.swing.JLabel playlistTotalLabel;
    private javax.swing.JButton sortButton;
    private javax.swing.JTextField titleTextField;
    // End of variables declaration//GEN-END:variables
}
```