

```
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)";
    }
}
```

```
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 pSong)
    {
        boolean found = false;
        int i = 0;
        while (!found && i < alSongs.size())
        {
            // autre version (meilleure, plus exacte):
            // pSong.getArtist().equals(s.getArtist()) && pSong.getTitle().equals(s.getTitle()) && pSong.getDuration() == s.getDuration()
            if (pSong.toString().equals(alSongs.get(i).toString()))
                found = true;
            else
                i++;
        }
        if (found)
            return false;
        else
            return alSongs.add(pSong);
    }

    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
```

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

```
public class Cd extends SongList
{
    private int capacity = 74 * 60;

    public Cd()
    {
        // capacité par défaut
    }

    public Cd(int pCapacity)
    {
        capacity = pCapacity;
    }

    public Cd(double pCapacity)
    {
        capacity = (int) (pCapacity * 60);
    }

    public int getCapacity()
    {
        return capacity;
    }

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

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

```

import javax.swing.JOptionPane;
public class MainFrame extends javax.swing.JFrame
{
    private SongList songList = new SongList();
    private Cd cd = new Cd(74.0);

    public MainFrame()
    {
        initComponents();

        // Ajouter des données de test... plus simple que de le faire à la main
        fillTestData();

        updateView();
    }

    public void fillTestData()
    {
        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));
    }

    public void updateView()
    {
        allSongsList.setListData(songList.toArray());
        cdList.setListData(cd.toArray());

        durationLabel.setText(songList.calculateDuration() + "s");
        cdDurationLabel.setText(cd.calculateDuration() + "s");
        cdFreeLabel.setText(cd.getFreeCapacity() + "s");
        cdTotalLabel.setText(cd.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

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

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

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

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

    // Suite... page suivante

```

```
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
// Skipped: ... graphic attributes
}
```