

```
public class Participant
{
    // Les informations nécessaires

    private String firstname;
    private String name;
    private int birthYear;

    public Participant(String pFirstname, String pName, int pBirthYear)
    {
        firstname = pFirstname;
        name      = pName;
        birthYear = pBirthYear;
    }

    public String getFirstname()
    {
        return firstname;
    }

    public String getName()
    {
        return name;
    }

    public int getBirthYear()
    {
        return birthYear;
    }

    public String toString()
    {
        return firstname + " " + name + " (" + birthYear + ")";
    }
}
```

```
import java.util.ArrayList;
public class Participants
{
    // Liste des participants au cours
    private ArrayList<Participant> alParticipants = new ArrayList<>();

    public void add(Participant pParticipant)
    {
        alParticipants.add(pParticipant);
    }

    public void remove(int i)
    {
        alParticipants.remove(i);
    }

    public int size()
    {
        return alParticipants.size();
    }

    public Participant get(int pPosition)
    {
        return alParticipants.get(pPosition);
    }

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

    public Participant getEldest()
    {
        Participant current, eldest;
        if (size() == 0)
            return null;

        eldest = alParticipants.get(0);
        for (int i = 1; i < size(); i++)
        {
            current = alParticipants.get(i);
            if (current.getBirthYear() < eldest.getBirthYear())
                eldest = current;
        }

        return eldest;
    }

    public Participant getYoungest()
    {
        Participant current, youngest;
        if (size() == 0)
            return null;

        youngest = alParticipants.get(0);
        for (int i = 1; i < size(); i++)
        {
            current = alParticipants.get(i);
            if (current.getBirthYear() > youngest.getBirthYear())
                youngest = current;
        }

        return youngest;
    }

    public int searchByBirthYear(int pYear)
    {
        boolean found = false;

        int i = 0;
        while (!found && (i < alParticipants.size()))
        {
            if (pYear == alParticipants.get(i).getBirthYear())
                found = true;
            else
                i++;
        }

        if (found)
            return i;
        else
            return -1;
    }

    // ... suite page suivante ...
}
```

```
public int searchByName(String pName)
{
    boolean found = false;

    int i = 0;
    while (!found && (i < alParticipants.size()))
    {
        if (pName.equals(alParticipants.get(i).getName()))
            found = true;
        else
            i++;
    }

    if (found)
        return i;
    else
        return -1;
}

public int countTextInFirstName(String text)
{
    int count = 0;

    for (int i = 0; i < alParticipants.size(); i++)
    {
        if (alParticipants.get(i).getFirstname().contains(text))
        {
            count++;
        }
    }

    return count;
}

public void removeOlder(int year)
{
    for (int i = alParticipants.size() - 1; i >= 0; i--)
    {
        if (alParticipants.get(i).getBirthYear() < year)
        {
            alParticipants.remove(i);
        }
    }
}

public Participant searchLastByName(String pName)
{
    boolean found = false;

    int i = alParticipants.size() - 1;
    while (!found && (i >= 0))
    {
        if (pName.equals(alParticipants.get(i).getName()))
            found = true;
        else
            i--;
    }

    if (found)
        return alParticipants.get(i);
    else
        return null;
}

public ArrayList<Participant> searchAllYounger(int pYear)
{
    ArrayList<Participant> res = new ArrayList<>();
    for (int i = 0; i < alParticipants.size(); i++)
    {
        if (alParticipants.get(i).getBirthYear() > pYear)
            res.add(alParticipants.get(i));
    }
    return res;
}

public int searchNthByYear(int n, int pYear)
{
    int count = 0;
    int pos = 0;

    int i = 0;
    while ((count < n) && (i < alParticipants.size()))
    {
        if (alParticipants.get(i).getBirthYear() == pYear)
        {
            count++;
            pos = i;
        }

        i++;
    }

    if (count == n)
        return pos;
    else
        return -1;
}
```

```
public Participant searchNthParticipantByYear(int n, int pYear)
{
    ArrayList<Participant> list = new ArrayList<>();
    for (int i = 0; i < alParticipants.size(); i++)
    {
        if (alParticipants.get(i).getBirthYear() == pYear)
            list.add(alParticipants.get(i));
    }

    if (list.size() >= n)
        return alParticipants.get(n);
    else
        return null;
}

public void sortByNameAsc()
{
    for (int i = 0; i < alParticipants.size() - 1; i++)
    {
        String min = alParticipants.get(i).getName();
        int posMin = i;
        for (int j = i + 1; j < alParticipants.size(); j++)
        {
            if (alParticipants.get(j).getName().compareTo(min) < 0)
            {
                min = alParticipants.get(j).getName();
                posMin = j;
            }
        }

        if (i != posMin)
        {
            Participant temp = alParticipants.get(i);
            alParticipants.set(i, alParticipants.get(posMin));
            alParticipants.set(posMin, temp);
        }
    }
}

public void sortByNameDesc()
{
    for (int i = 0; i < alParticipants.size() - 1; i++)
    {
        String max = alParticipants.get(i).getName();
        int posMax = i;
        for (int j = i + 1; j < alParticipants.size(); j++)
        {
            if (alParticipants.get(j).getName().compareTo(max) > 0)
            {
                max = alParticipants.get(j).getName();
                posMax = j;
            }
        }

        if (i != posMax)
        {
            Participant temp = alParticipants.get(i);
            alParticipants.set(i, alParticipants.get(posMax));
            alParticipants.set(posMax, temp);
        }
    }
}
```

```
// ... suite page suivante ...
```

```
public void sortByYearAsc()
{
    for (int i = 0; i < alParticipants.size() - 1; i++)
    {
        int min = alParticipants.get(i).getBirthYear();
        int posMin = i;
        for (int j = i + 1; j < alParticipants.size(); j++)
        {
            if (alParticipants.get(j).getBirthYear() < min)
            {
                min = alParticipants.get(j).getBirthYear();
                posMin = j;
            }
        }
        if (i != posMin)
        {
            Participant temp = alParticipants.get(i);
            alParticipants.set(i, alParticipants.get(posMin));
            alParticipants.set(posMin, temp);
        }
    }
}

public void sortByYearDesc()
{
    for (int i = 0; i < alParticipants.size() - 1; i++)
    {
        int max = alParticipants.get(i).getBirthYear();
        int posMax = i;
        for (int j = i + 1; j < alParticipants.size(); j++)
        {
            if (alParticipants.get(j).getBirthYear() > max)
            {
                max = alParticipants.get(j).getBirthYear();
                posMax = j;
            }
        }
        if (i != posMax)
        {
            Participant temp = alParticipants.get(i);
            alParticipants.set(i, alParticipants.get(posMax));
            alParticipants.set(posMax, temp);
        }
    }
}
```

```

import java.util.ArrayList;
public class MainFrame extends javax.swing.JFrame
{
    private Participants participants = new Participants();

    public MainFrame()
    {
        initComponents();

        // TEST PURPOSES ONLY...
        participants.add(new Participant("Josy", "Muller", 1980));
        participants.add(new Participant("Ren", "Klein", 1966));
        participants.add(new Participant("Poli", "Klein", 1985));
        participants.add(new Participant("Mary", "Nemo", 1978));

        updateView();
    }

    private void updateView()
    {
        participantsList.setListData(participants.toArray());

        // afficher un "-" si "eldest" ou "youngest" n'existe pas
        Participant eldest = participants.getEldest();
        if (eldest == null)
            eldestLabel.setText("-");
        else
            eldestLabel.setText(eldest.toString());

        Participant youngest = participants.getYoungest();
        if (youngest == null)
            youngestLabel.setText("-");
        else
            youngestLabel.setText(youngest.toString());
    }
// Skipped: ... initComponents { ... }
    private void addButtonActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_addButtonActionPerformed
        String firstname = firstnameTextField.getText();
        String name = nameTextField.getText();
        int year = Integer.valueOf(yearTextField.getText());

        Participant p = new Participant(firstname, name, year);
        participants.add(p);

        // Efface tous les champs
        nameTextField.setText("");
        firstnameTextField.setText("");
        yearTextField.setText("");

        // Améliorations: remet le curseur dans le champ 'prénom'
        firstnameTextField.requestFocus();

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

    private void removeButtonActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_removeButtonActionPerformed
        // Vérifier s'il y a bien qqchose de sélectionné
        int index = participantsList.getSelectedIndex();
        if (index >= 0)
            participants.remove(index);
        updateView();
    } //GEN-LAST:event_removeButtonActionPerformed

    private void searchBirthyearButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_searchBirthyearButtonActionPerformed
    { //GEN-HEADEREND:event_searchBirthyearButtonActionPerformed
        // désélectionne la liste
        participantsList.clearSelection();

        String yearVal = yearTextField.getText();
        if (yearVal.equals(""))
            return;

        int year = Integer.valueOf(yearVal);
        int pos = participants.searchByBirthYear(year);
        if (pos != -1)
            participantsList.setSelectedIndex(pos);
    } //GEN-LAST:event_searchBirthyearButtonActionPerformed

    private void searchNameButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_searchNameButtonActionPerformed
    { //GEN-HEADEREND:event_searchNameButtonActionPerformed
        // désélectionne la liste
        participantsList.clearSelection();

        String name = nameTextField.getText();
        if (name.equals(""))
            return;

        int pos = participants.searchByName(name);
        if (pos != -1)
            participantsList.setSelectedIndex(pos);
    } //GEN-LAST:event_searchNameButtonActionPerformed

    // ... suite page suivante ...

```

```

private void countFirstnameButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_countFirstnameButtonActionPerformed
{ //GEN-HEADEREND:event_countFirstnameButtonActionPerformed
    // enlève le compteur s'il existait avant
    countLabel.setText("-");

    String firstName = firstNameTextField.getText();
    if (firstName.equals(""))
        return;
    int count = participants.countTextInFirstName(firstName);
    countLabel.setText("Count: " + count);
} //GEN-LAST:event_countFirstnameButtonActionPerformed

private void removeOlderButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_removeOlderButtonActionPerformed
{ //GEN-HEADEREND:event_removeOlderButtonActionPerformed
    String yearVal = yearTextField.getText();
    if (yearVal.equals(""))
        return;

    int year = Integer.valueOf(yearVal);
    participants.removeOlder(year);
    updateView();
} //GEN-LAST:event_removeOlderButtonActionPerformed

private void findLastByNameButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_findLastByNameButtonActionPerformed
{ //GEN-HEADEREND:event_findLastByNameButtonActionPerformed
    // enlève le résultat s'il existait avant
    lastLabel.setText("-");

    String name = nameTextField.getText();
    if (name.equals(""))
        return;

    Participant p = participants.searchLastByName(name);
    if (p != null)
        lastLabel.setText(p.toString());
} //GEN-LAST:event_findLastByNameButtonActionPerformed

private void findAllButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_findAllButtonActionPerformed
{ //GEN-HEADEREND:event_findAllButtonActionPerformed
    // enlève le résultat s'il existait avant
    nbrAllLabel.setText("-");

    String yearVal = yearTextField.getText();
    if (yearVal.equals(""))
        return;

    int year = Integer.valueOf(yearVal);
    ArrayList<Participant> res = participants.searchAllYounger(year);
    nbrAllLabel.setText("Number of participants found: " + res.size());
} //GEN-LAST:event_findAllButtonActionPerformed

private void findNthButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_findNthButtonActionPerformed
{ //GEN-HEADEREND:event_findNthButtonActionPerformed
    // désélectionne la liste
    participantsList.clearSelection();

    String yearVal = yearTextField.getText();
    String nVal = nthTextField.getText();

    if (!yearVal.equals("") && !nVal.equals(""))
    {
        int year = Integer.valueOf(yearVal);
        int n = Integer.valueOf(nVal);
        int pos = participants.searchNthByYear(n, year);
        if (pos != -1)
            participantsList.setSelectedIndex(pos);
    }
} //GEN-LAST:event_findNthButtonActionPerformed

private void nthParticipantButtonActionPerformed(java.awt.event.ActionEvent evt) //GEN-FIRST:event_nthParticipantButtonActionPerformed
{ //GEN-HEADEREND:event_nthParticipantButtonActionPerformed
    // enlève le résultat s'il existait avant
    nthParticipantLabel.setText("-");

    String yearVal = yearTextField.getText();
    String nVal = nthParticipantTextField.getText();

    if (!yearVal.equals("") && !nVal.equals(""))
    {
        int year = Integer.valueOf(yearVal);
        int n = Integer.valueOf(nVal);
        Participant p = participants.searchNthParticipantByYear(n, year);
        if (p != null)
            nthParticipantLabel.setText(String.valueOf(p));
    }
} //GEN-LAST:event_nthParticipantButtonActionPerformed

// ... suite page suivante ...

```

```

private void sortByNameASCButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_sortByNameASCButtonActionPerformed
    participants.sortByNameAsc();
    updateView();
}//GEN-LAST:event_sortByNameASCButtonActionPerformed

private void sortByNameDESCButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_sortByNameDESCButtonActionPerformed
    participants.sortByNameDesc();
    updateView();
}//GEN-LAST:event_sortByNameDESCButtonActionPerformed

private void sortByYearASCButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_sortByYearASCButtonActionPerformed
    participants.sortByYearAsc();
    updateView();
}//GEN-LAST:event_sortByYearASCButtonActionPerformed

private void sortByYearDESCButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_sortByYearDESCButtonActionPerformed
    participants.sortByYearDesc();
    updateView();
}//GEN-LAST:event_sortByYearDESCButtonActionPerformed
// Skipped: ... Look @ Feel
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton addButton;
private javax.swing.JButton countFirstnameButton;
private javax.swing.JLabel countLabel;
private javax.swing.JLabel eldestLabel;
private javax.swing.JButton findAllButton;
private javax.swing.JButton findLastByNameButton;
private javax.swing.JButton findNthButton;
private javax.swing.JTextField firstnameTextField;
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.JScrollPane jScrollPane1;
private javax.swing.JLabel lastLabel;
private javax.swing.JTextField nameTextField;
private javax.swing.JLabel nbrAllLabel;
private javax.swing.JButton nthParticipantButton;
private javax.swing.JLabel nthParticipantLabel;
private javax.swing.JTextField nthParticipantTextField;
private javax.swing.JTextField nthTextField;
private javax.swing.JList participantsList;
private javax.swing.JButton removeButton;
private javax.swing.JButton removeOlderButton;
private javax.swing.JButton searchBirthyearButton;
private javax.swing.JButton searchNameButton;
private javax.swing.JButton sortByNameASCButton;
private javax.swing.JButton sortByNameDESCButton;
private javax.swing.JButton sortByYearASCButton;
private javax.swing.JButton sortByYearDESCButton;
private javax.swing.JTextField yearTextField;
private javax.swing.JLabel youngestLabel;
// End of variables declaration//GEN-END:variables
}

```