**Michal Bochenek  
EC1401916**

**Technical Manual**

**For Assessment on Sets**

**Table of contents:  
  
1. Program Specification.  
2. Code Listing.  
3. Testing.**

1. **Program Specification.**   
     
   Program performs operations on two sets containing names of top 10 chess players from year 2000 and 2015. The result is being displayed on screen and saved to a textfile called ChessResults.  
     
   **List of operations to perform:**1. Display all the players from chess2000  
   2. Display all the players from chess2015  
   3. Display the player’s common to both sets. (Intersection)  
   4. Display all the players from both sets. (Union)  
   5. Display all the players in chess2000 that are not in chess2015. (Difference)  
   6. Display all the players in chess2015 that are not in chess2000. (Difference)  
   7. Add a duplicate player and show the set has not changed.  
     
   **Example of a working program:**Console:

|  |
| --- |
| Players from the list 2015:  [[Kramnik, Vladimir], [Topalov, Veselin], [Aronian, Levon], [Giri, Anish], [Carlsen, Magnus], [Nakamura, Hikaru], [Anand, Viswanathan], [Caruana, Fablano], [So, Wesley], [Grischuk, Alexander]]  Players from the list 2000:  [[Leko, Peter], [Kramnik, Vladimir], [Shirov, Alexei], [Topalov, Veselin], [Morozevich, Alexander], [Kasparov, Garry], [Adams, Michael], [Ivanchuk, Vassily], [Anand, Viswanathan], [Krasenkow, Michal]]  TASK 3 Players common to both sets:  [[Kramnik, Vladimir], [Topalov, Veselin], [Anand, Viswanathan]] |

Textfile:

|  |
| --- |
| Task1: Display all players from chess2000 set.  [Kramnik, Vladimir]  [Topalov, Veselin]  [Aronian, Levon]  [Giri, Anish]  [Carlsen, Magnus]  [Nakamura, Hikaru]  [Anand, Viswanathan]  [Caruana, Fablano]  [So, Wesley]  [Grischuk, Alexander]  Task2: Display all players from chess2015 set.  [Leko, Peter]  [Kramnik, Vladimir]  [Shirov, Alexei]  [Topalov, Veselin]  [Morozevich, Alexander]  [Kasparov, Garry]  [Adams, Michael]  [Ivanchuk, Vassily]  [Anand, Viswanathan]  [Krasenkow, Michal] |

1. **Code Listing.**  
     
   **2.1 Main program code:**

package chesssets;

import java.io.BufferedWriter;

import java.io.File;

import java.util.Set;

import java.util.HashSet;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

/\*\*

\*

\* @author Michal Bochenek

\*/

public class ChessSets

{

public static void main(String[] args)

{

//create new hash set for players from May 2015

Set<String> chess2015 = new HashSet<>();

//create new hash set for players from July 2000

Set<String> chess2000 = new HashSet<>();

//create new hash set for TASK 3

Set<String> task3 = new HashSet<>();

//create new hash set for TASK 4

Set<String> task4 = new HashSet<>();

//create new hash set for TASK 5

Set<String> task5 = new HashSet<>();

//create new hash set for TASK 6

Set<String> task6 = new HashSet<>();

//create new hash set for TASK 7

Set<String> task7 = new HashSet<>();

//add players from the lists to the sets ([Surname, Forename])

chess2015.add("[Carlsen, Magnus]");

chess2015.add("[Anand, Viswanathan]");

chess2015.add("[Caruana, Fablano]");

chess2015.add("[Nakamura, Hikaru]");

chess2015.add("[Topalov, Veselin]");

chess2015.add("[Grischuk, Alexander]");

chess2015.add("[So, Wesley]");

chess2015.add("[Kramnik, Vladimir]");

chess2015.add("[Giri, Anish]");

chess2015.add("[Aronian, Levon]");

chess2000.add("[Kasparov, Garry]");

chess2000.add("[Kramnik, Vladimir]");

chess2000.add("[Anand, Viswanathan]");

chess2000.add("[Morozevich, Alexander]");

chess2000.add("[Adams, Michael]");

chess2000.add("[Shirov, Alexei]");

chess2000.add("[Leko, Peter]");

chess2000.add("[Ivanchuk, Vassily]");

chess2000.add("[Topalov, Veselin]");

chess2000.add("[Krasenkow, Michal]");

//begin writing tasks to the text file and display results on screen

//TASK 1 and 2: display both sets:

System.out.println("Players from the list 2015: ");

System.out.println("");

System.out.println(chess2015);

System.out.println("");

System.out.println("Players from the list 2000: ");

System.out.println("");

System.out.println(chess2000);

System.out.println("");

//TASK 3 Display the player's common to both sets (INTERSECTION):

System.out.println("TASK 3 Players common to both sets: ");

System.out.println("");

task3 = SFunction.intersection(chess2015, chess2000);

System.out.println(task3);

System.out.println("");

//TASK 4 Display all the players from both sets. (UNION)

System.out.println("TASK 4 Players from both sets: ");

System.out.println("");

task4 = SFunction.union(chess2015, chess2000);

System.out.println(task4);

System.out.println("");

//TASK 5 Display all the players in chess2000 that are not in chess2015

System.out.println("TASK 5 Display all the players in chess2000 that are not in chess 2015: ");

System.out.println("");

task5 = SFunction.difference(chess2000, chess2015);

System.out.println(task5);

System.out.println("");

//TASK 6 Display all the players in chess2015 that are not in chess2000

System.out.println("TASK 6 Display all the players in chess2015 that are not in chess2000: ");

System.out.println("");

task6 = SFunction.difference(chess2015, chess2000);

System.out.println(task6);

System.out.println("");

//TASK 7 Add a duplicate player and show the set has not changed

System.out.println("TASK 7 Add a duplicate player and show the set has not changed: ");

//note: I use task7 to not change main set chess2015, to avoid errors

task7.addAll(chess2015);

task7.add("[Carlsen, Magnus]");

System.out.println("We have added [Carlsen, Magnus] to the set chess2015.");

System.out.println("Set chess2015 size before addition: "+chess2015.size());

System.out.println("Set 2015 size after addition of existing name: "+task7.size());

System.out.println("");

//bool variable for later use in writing to file

boolean setsize;

if(task7.size()==chess2015.size())

{

System.out.println("Set didn't change sizes. It can contain only one element of each.");

setsize = true;

}

else

{

System.out.println("Set did change. It contains more than one element of each.");

setsize = false;

}

//create text file ChessResults to store the task answers

File chess = new File("ChessResults.txt");

//create string arrays for use in writing to file

String[] chess15 = new String[chess2015.size()];

String[] chess00 = new String[chess2000.size()];

//create string arrays for the results [Use size of - as we can't predict current set size]

String[] t3 = new String[task3.size()];

String[] t4 = new String[task4.size()];

String[] t5 = new String[task5.size()];

String[] t6 = new String[task6.size()];

//put sets content to the appropriate arrays

chess2015.toArray(chess15);

chess2000.toArray(chess00);

task3.toArray(t3);

task4.toArray(t4);

task5.toArray(t5);

task6.toArray(t6);

try

{

BufferedWriter writer = new BufferedWriter(new FileWriter(chess));

PrintWriter print = new PrintWriter(writer);

String newline = System.getProperty("line.separator");

//TASK 1

writer.write("Task1: Display all players from chess2000 set. ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int i=0; i<10; i++)

{

writer.write(chess15[i]);

writer.write(newline);

}

//TASK 2

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("Task2: Display all players from chess2015 set. ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int j=0; j<10; j++)

{

writer.write(chess00[j]);

writer.write(newline);

}

//TASK 3

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("Task3: Players common to both sets: ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int k=0; k<t3.length; k++)

{

writer.write(t3[k]);

writer.write(newline);

}

//TASK 4

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("Task4: All players from both sets: ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int l=0; l<t4.length; l++)

{

writer.write(t4[l]);

writer.write(newline);

}

//TASK 5

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("Task5: All players in chess2000 that are not in chess2015: ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int m=0; m<t5.length; m++)

{

writer.write(t5[m]);

writer.write(newline);

}

//TASK 6

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("Task6: All players in chess2015 that are not in chess2000: ");

writer.write(newline);

writer.write("");

writer.write(newline);

for(int n=0; n<t6.length; n++)

{

writer.write(t6[n]);

writer.write(newline);

}

//TASK 7

writer.write(newline);

writer.write("");

writer.write(newline);

writer.write("TASK 7 Add a duplicate player and show the set has not changed: ");

writer.write(newline);

writer.write("");

writer.write("Add: [Carlsen, Magnus] to the set chess2015");

writer.write(newline);

writer.write("");

writer.write("Set size did not change: "+setsize);

writer.write(newline);

writer.write("END OF FILE, ALL TASKS COMPLETED");

//Close writer

writer.close();

}

catch(IOException e)

{

System.out.println("An unknown error has occured!");

}

}

}

**2.2 SFunction class content:**  
  
package chesssets;

import java.util.Set;

import java.util.HashSet;

/\*\*

\*

\* @author Michal Bochenek

\*/

public class SFunction

{

public static <T> Set<T> union(Set<T> setA, Set<T> setB)

{

Set<T> tmp = new HashSet<T>(setA);

tmp.addAll(setB);

return tmp;

}

public static <T> Set<T> intersection(Set<T> setA, Set<T> setB)

{

Set<T> tmp = new HashSet<T>();

for (T x : setA)

if (setB.contains(x))

tmp.add(x);

return tmp;

}

public static <T> Set<T> complement(Set<T> setA, Set<T> setB)

{

Set<T> tmp = new HashSet<T>();

for (T x : setA)

if (!setB.contains(x))

tmp.add(x);

return tmp;

}

public static <T> Set<T> difference(Set<T> setA, Set<T> setB)

{

Set<T> tmp = new HashSet<T>(setA);

tmp.removeAll(setB);

return tmp;

}

public static <T> Set<T> symDifference(Set<T> setA, Set<T> setB)

{

Set<T> tmpA;

Set<T> tmpB;

tmpA = union(setA, setB);

tmpB = intersection(setA, setB);

return difference(tmpA, tmpB);

}

public static <T> boolean isSubset(Set<T> setA, Set<T> setB)

{

return setB.containsAll(setA);

}

public static <T> boolean isSuperset(Set<T> setA, Set<T> setB)

{

return setA.containsAll(setB);

}

}

1. **Testing (White box method):**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test nr:** | **Details:** | **Result:** | **Date:** |
| 1. | Displaying console result: run:  Players from the list 2015:  [[Kramnik, Vladimir], [Topalov, Veselin], [Aronian, Levon], [Giri, Anish], [Carlsen, Magnus], [Nakamura, Hikaru], [Anand, Viswanathan], [Caruana, Fablano], [So, Wesley], [Grischuk, Alexander]]  Players from the list 2000:  [[Leko, Peter], [Kramnik, Vladimir], [Shirov, Alexei], [Topalov, Veselin], [Morozevich, Alexander], [Kasparov, Garry], [Adams, Michael], [Ivanchuk, Vassily], [Anand, Viswanathan], [Krasenkow, Michal]]  TASK 3 Players common to both sets:  [[Kramnik, Vladimir], [Topalov, Veselin], [Anand, Viswanathan]]  TASK 4 Players from both sets:  [[Kramnik, Vladimir], [Shirov, Alexei], [Morozevich, Alexander], [Giri, Anish], [Carlsen, Magnus], [Ivanchuk, Vassily], [Caruana, Fablano], [Grischuk, Alexander], [Leko, Peter], [Topalov, Veselin], [Aronian, Levon], [Kasparov, Garry], [Adams, Michael], [Nakamura, Hikaru], [Anand, Viswanathan], [Krasenkow, Michal], [So, Wesley]]  TASK 5 Display all the players in chess2000 that are not in chess 2015:  [[Leko, Peter], [Shirov, Alexei], [Morozevich, Alexander], [Kasparov, Garry], [Adams, Michael], [Ivanchuk, Vassily], [Krasenkow, Michal]]  TASK 6 Display all the players in chess2015 that are not in chess2000:  [[Aronian, Levon], [Giri, Anish], [Carlsen, Magnus], [Nakamura, Hikaru], [Caruana, Fablano], [So, Wesley], [Grischuk, Alexander]]  TASK 7 Add a duplicate player and show the set has not changed:  We have added [Carlsen, Magnus] to the set chess2015.  Set chess2015 size before addition: 10  Set 2015 size after addition of existing name: 10  Set didn't change sizes. It can contain only one element of each.  BUILD SUCCESSFUL (total time: 0 seconds) | OK  program displays all results in the matter they were coded | 19/11/2015 |
| 2. | Textfile content:  Task1: Display all players from chess2000 set.  [Kramnik, Vladimir]  [Topalov, Veselin]  [Aronian, Levon]  [Giri, Anish]  [Carlsen, Magnus]  [Nakamura, Hikaru]  [Anand, Viswanathan]  [Caruana, Fablano]  [So, Wesley]  [Grischuk, Alexander]  Task2: Display all players from chess2015 set.  [Leko, Peter]  [Kramnik, Vladimir]  [Shirov, Alexei]  [Topalov, Veselin]  [Morozevich, Alexander]  [Kasparov, Garry]  [Adams, Michael]  [Ivanchuk, Vassily]  [Anand, Viswanathan]  [Krasenkow, Michal]  Task3: Players common to both sets:  [Kramnik, Vladimir]  [Topalov, Veselin]  [Anand, Viswanathan]  Task4: All players from both sets:  [Kramnik, Vladimir]  [Shirov, Alexei]  [Morozevich, Alexander]  [Giri, Anish]  [Carlsen, Magnus]  [Ivanchuk, Vassily]  [Caruana, Fablano]  [Grischuk, Alexander]  [Leko, Peter]  [Topalov, Veselin]  [Aronian, Levon]  [Kasparov, Garry]  [Adams, Michael]  [Nakamura, Hikaru]  [Anand, Viswanathan]  [Krasenkow, Michal]  [So, Wesley]  Task5: All players in chess2000 that are not in chess2015:  [Leko, Peter]  [Shirov, Alexei]  [Morozevich, Alexander]  [Kasparov, Garry]  [Adams, Michael]  [Ivanchuk, Vassily]  [Krasenkow, Michal]  Task6: All players in chess2015 that are not in chess2000:  [Aronian, Levon]  [Giri, Anish]  [Carlsen, Magnus]  [Nakamura, Hikaru]  [Caruana, Fablano]  [So, Wesley]  [Grischuk, Alexander]  TASK 7 Add a duplicate player and show the set has not changed:  Add: [Carlsen, Magnus] to the set chess2015  Set size did not change: true  END OF FILE, ALL TASKS COMPLETED | OK  program writes to file all required data and saves it in readable format | 19/11/2015 |
| 3. | Error handling (removed file name ChessResult.txt)   File chess = new File("");  Result: Console displays:  An unknown error has occured! | OK  error handling displays information to the user | 19/11/2015 |