

PMR: This project was also relatively easy since it was built on the previous 2. I enjoyed this project. I look forward to more projects built on top of each other.

HOMEWORK3

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public abstract class Homework3 implements Processing, Comparable<Homework3>

{

//variables

public int pagesRead;

public String typeHomework;

//constructor

Homework3()

{

pagesRead = 0;

typeHomework = "none";

}

//other methods

//create assignment

public abstract void createAssignment(int p);

//compare

public int compareTo(Homework3 homework)

{

if(this.pagesRead < homework.pagesRead)

{

return -1;

}

else if(this.pagesRead == homework.pagesRead)

{

return 0;

}

else

{

return 1;

}

}

}

MYENGLISH3

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public class MyEnglish3 extends Homework3

{

//constructor

MyEnglish3()

{

super();

}

//implemented create assignment method

public void createAssignment(int p)

{

pagesRead = p;

typeHomework = "English";

}

//other methods

//to string

public String toString()

{

return typeHomework + " - must read " + pagesRead + " pages";

}

//implement doReading()

public void doReading()

{

pagesRead -= 1;

}

}

MYJAVA3

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public class MyJava3 extends Homework3

{

//constructor

MyJava3()

{

super();

}

//implemented create assignment

public void createAssignment(int p)

{

pagesRead = p;

typeHomework = "Java";

}

//other methods

//to string

public String toString()

{

return typeHomework + " - must read " + pagesRead + " pages";

}

//implements doReading()

public void doReading()

{

pagesRead -= 4;

}

}

MYMATH3

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public class MyMath3 extends Homework3

{

//constructor

MyMath3()

{

super();

}

//implemented create assignment method

public void createAssignment(int p)

{

pagesRead = p;

typeHomework = "Math";

}

//other methods

//to string

public String toString()

{

return typeHomework + " - must read " + pagesRead + " pages";

}

//implement doReading

public void doReading()

{

pagesRead -= 2;

}

}

MYSCIENCE3

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public class MyScience3 extends Homework3

{

//constructor

MyScience3()

{

super();

}

//implemented create assignment method

public void createAssignment(int p)

{

pagesRead = p;

typeHomework = "Science";

}

//other methods

//to string

public String toString()

{

return typeHomework + " - must read " + pagesRead + " pages";

}

//implements doReading()

public void doReading()

{

pagesRead -= 3;

}

}

PROCESSING

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: add an interface to process pages from your homework

\*

\* PMR in README.txt

\*/

public interface Processing

{

//methods

void doReading();

}

TESTHOMEWORK3

import java.util.\*;

/\*

\* By Anika Jallipalli

\* Date: 3/1/2020

\* Purpose: implement the Comparable<T> interface

\*

\* PMR in README.txt

\*/

public class TestHomework3

{

public static void main(String[] args)

{

//create the arraylist

List<Homework3> homework = new ArrayList<Homework3>();

//create each homework element

MyMath3 math = new MyMath3();

math.createAssignment(4);

MyScience3 science = new MyScience3();

science.createAssignment(6);

MyEnglish3 english = new MyEnglish3();

english.createAssignment(4);

MyJava3 java = new MyJava3();

java.createAssignment(5);

//add each homework element

homework.add(math);

homework.add(science);

homework.add(english);

homework.add(java);

//print out results

for(Homework3 assignment : homework)

{

System.out.println(assignment);

}

//sort the homework from least to greatest reading

Collections.sort(homework);

System.out.println();

System.out.println("Sorted Homework");

//print out sorted homework

for(Homework3 assignment : homework)

{

System.out.println(assignment);

}

//check for assignments with equal value

System.out.println();

System.out.println("Equal sized assignments");

List<String> combos = new ArrayList<String>();

for(int i = 0; i < homework.size(); i++)

{

for(int j = 0; j < homework.size(); j++)

{

if(j != i)

{

if(!combos.contains(homework.get(i) + " " + homework.get(j)) && !combos.contains(homework.get(j) + " " + homework.get(i)))

{

if(homework.get(i).compareTo(homework.get(j)) == 0)

{

System.out.println("The homework for " + homework.get(i).typeHomework + " and " + homework.get(j).typeHomework + " are the same number of pages");

combos.add(homework.get(i) + " " + homework.get(j));

}

}

}

}

}

}

}