```java
//Professor Ziegler
//HW9
//Jinglin Tan

import java.util.Scanner;   //needed to use Scanner
import java.io.*;   //needed to use PrintWriter

public class pgm9 {

    public static void main(String[] args) throws IOException{
        final int MAX = 50;     //max size of arrays
        int donorcount;         //to store number of set of read data
        int[] idnumbers = new int[MAX];     //array of donor ids
        int[] donations = new int[MAX];     //array of donations

        File myfile = new File("c:/myinput.txt");   //create a file object
        //Scanner input = new Scanner(System.in);
        Scanner input = new Scanner(myfile);        //read from file

        //PrintWriter output = new PrintWriter(System.out);
        PrintWriter output = new PrintWriter("c:/myoutput.txt");  //write to file

        //call method readData() and store size of set of read data
        donorcount = readData(idnumbers, donations, input);

        //call method print() to print the original set of data into table
        print(idnumbers, donations, donorcount, output);

        //call method sortId() to sort ID numbers into numerical order
        sortId(idnumbers, donations, donorcount);

        //call print() to print the sorted data
        print(idnumbers, donations, donorcount, output);

        //call method sortDonation() to sort donations into numerical order
        sortDonation(idnumbers, donations, donorcount);

        //call print() to print the sorted data
        print(idnumbers, donations, donorcount, output);

        output.flush();     //flush the buffer
        input.close();      //close input file
        output.close();     //close output file
    }

    /*method readData()
     * input: id - array of integers to store id numbers
     *        dona - array of integers to store donations
     *        input - Scanner object
     * process: read sets of data from input file into arrays until there is
     *          nothing to read
     *          count the sets of data
     * output: return the number of sets of data(size)
     */
    public static int readData(int[] id, int[] dona, Scanner input){
        int size = 0;       //to count the sets of read data
        while(input.hasNext()){
            id[size] = input.nextInt();     //read data to array of id
            dona[size] = input.nextInt();   //read data to array of dona
            size++;
        }
```

```
    return size;    //return size
}

/*method print()
 * input: id - array of integers of id numbers
 *        dona - array of integers of donations
 *        size - number of data in an array
 *        output - PrintWriter object
 * process: Print sets of data in arrays of id and dona into a table
 * output: Print sets of data in arrays of id and dona into a table
 */
public static void print(int[] id, int[] dona, int size, PrintWriter output){
    output.println(" Table of donations");  //overall heading
    output.println();
    output.println(" ID\t Donations");  //headings
    for(int i = 0; i < size; i++){
        output.printf(" %d%11d", id[i], dona[i]);   //print ID and donation
        output.println();
    }
    output.println();
}

/*method sortId()
 * input: id - array of integers of id numbers
 *        dona - array of integers of donations
 *        size - number of data in an array
 * process: use linear sort to sort ID numbers into numerical order maintaining
 *          match-up of ID numbers and donations
 * output: the sorted arrays
 */
public static void sortId(int[] id, int[] dona, int size){
    int temp;       //to store a temporary value to help exchange two values
    for(int i = 0; i < size - 1; i++){
        for(int j = i + 1; j < size; j++){
            if(id[i] > id[j]){
                temp = id[i];
                id[i] = id[j];
                id[j] = temp;
                temp = dona[i];
                dona[i] = dona[j];
                dona[j] = temp;
            }
        }
    }
}

/*method sortDonation()
 * input: id - array of integers of id numbers
 *        dona - array of integers of donations
 *        size - number of data in an array
 * process: use bubble sort to sort donations into numerical order maintaining
 *          the match-up of ID numbers and donations
 * output: the sorted arrays
 */
public static void sortDonation(int[] id, int[] dona, int size){
    int temp;       //to store a temporary value to help exchange two values
    boolean b = true;   //when b is true, continue the sort
    while(b == true){
        b = false;      //set it to false before the for loop
        for(int i = 0; i < size - 1; i++){
            if(dona[i] > dona[i + 1]){
```

```
                    temp = id[i];
                    id[i] = id[i + 1];
                    id[i + 1] = temp;
                    temp = dona[i];
                    dona[i] = dona[i + 1];
                    dona[i + 1] = temp;
                    b = true;   //it becomes true when there is an exchange of data
                }
            }
        }
    }


}
```