/\*

\* Aaron Knestaut

\* 10.26.15

\* Period A

\*

\* \*\*\*\*\*Program Description\*\*\*\*\*

\* The program generates 500 random numbers between 1 and 500 and replaces duplicates,

\* tries to find a user inputted value, then displays 4 numbers from the array.

\* \*\*\*\*\*Variable Dictionary\*\*\*\*\*

\* int a - holds static value to be tested against in the array

\* int b - temporarily holds a value to check against int a

\* int d - holds current dynamic value

\* int count - keeps track of how many numbers have been generated

\* boolean found - keeps track of if the value has been found

\* int lb - holds current lower bound

\* boolean nodupes - keeps track of if there are still duplicates in the array

\* int numbers [] - array for holding all values

\* int probe - int for browsing array numbers for int value

\* int s - holds current static value

\* int ub - holds the upper bound

\* int value - the user inputted number put into an int

\* String wanttofind - the user inputted number to be found

\*/

import javax.swing.JOptionPane;

public class BinarySearch

{

public static void main (String args [])

{

int numbers [] = new int [501];//declares/allocates the array

generator (numbers);//runs generator method

sorter (numbers);//runs sorter method

String wanttofind = JOptionPane.showInputDialog (null, "Input the number you want to search for.");//user inputs the value they want to find

int value = Integer.parseInt (wanttofind);//turns user inputted number into an int

search (numbers, value);//runs searcher method

JOptionPane.showMessageDialog (null, "Value 1: " + numbers[1] + "\nValue 251: " + numbers[251] +

"\nValue 337: " + numbers[337] + "\nValue 500: " + numbers[500]);//displays numbers at end

}//end of main method

public static void generator (int numbers [])

{

int count = 1;

while (count <= 500)//runs loop to generate numbers

{

numbers [count] = (int) (Math.random () \* 1000) + 1;

count = count + 1;

}//end of generator loop

}//end of generator method

public static void sorter (int numbers [])

{

boolean nodupes = false;

for (int s = 1; s < 500; s++)//runs a selection sort of the data

{

for (int d = s + 1; d <= 500; d++)

{

if(numbers [s] > numbers [d])

{

numbers [0] = numbers [d];

numbers [d] = numbers [s];

numbers [s] = numbers [0];

}

}

}

while (!nodupes)//gets rid of duplicates in data

{

for (int a = 0; a < numbers.length; a++)

{

for (int b = a + 1; b < numbers.length; b++)

{

if (b != a && numbers[b] == numbers[a])//checks if b is a duplicate to a

numbers[b] = (int) (Math.random () \* 1000) + 1;//generates a new random value for duplicate number

else

nodupes = true;//if there are no duplicates, the loop stops

}

}

for (int s = 1; s < 500; s++)//runs a selection sort of the deduped data

{

for (int d = s + 1; d <= 500; d++)

{

if(numbers [s] > numbers [d])

{

numbers [0] = numbers [d];

numbers [d] = numbers [s];

numbers [s] = numbers [0];

}

}

}

}

}//end of dedupe/sorter method

public static void search (int numbers [], int value)

{

int lb = 1;

int ub = 500;

boolean found = false;

int probe = (lb + ub)/2;

while (!found && ub > lb)//runs search loop only while found is not true and the upper bound is greater than the lower bound

{

if (value == numbers[probe])//if int probe is = to the value

found = true;//loop stops

else if (value > numbers[probe])

lb = probe + 1;

else

ub = probe - 1;

probe = (lb + ub)/2;//creates new probe

}//end of searcher loop

if (found == true)//tells if e value was found

JOptionPane.showMessageDialog (null, "Value found!");

if (ub <= lb)//tells if the value wasn't found

JOptionPane.showMessageDialog (null, "The value does not exist.");

}//end of search method

}