Last name First name, if group, Last2, First2	
class	CS3100
Assignment name/number	hwk2
Date submitted	“2014/04/11
Cs-login-name	
Hours programming since the course started	

This is type T2 hwks. Read on my website how to submit hwks. Ask me if something is not clear. Read this document carefully twice from start to end before doing any programming. All programs in this course use OOP. Read the programs on help 1500 and help2500 and help3100 and write some of those programs by yourself. Use 7SP.

For all hwk:
You must have all the classes, attributes, and methods I ask for. But, you can make additional attributes and methods if you feel you need them. No additional classes.
If I give you name of a class, signature of a method, or name of a variable, you must use the name exactly. Signature of method means it's name and type of any arguments.
If I let you select the arguments for your method, I will say doFunction(…). You can then pick any arguments you want or have no arguments if you choose so.
Similarly for return type, sometimes I will specify, sometimes I will let you choose.

In this course we may sometimes specify packages. If I don't specify package you use no package. (That is called in Java using default package.)

You will be graded on following the given requirements, technical elegance of your solution, satisfaction of requirements, program style and correctness.

Look at program ProgramStyle2500.java that does not do much but shows the program style we will be using in this course.

What is unspecified you can choose.

Any questions … ask in the class if you think it may benefit others, if it is specific to you – you can use the email or my office hours.
Hint:
Start early.

Do not submit data files, I will use my own.

Problem 1:
Write program BinSearch. Default package.

This program will read file data.txt which will reside in the same directory as the compiled code. The data file will contain sorted list of integers.

1. The program will have class BinSearch.
2. It will have object scope array of ints called arr of size 20.
3. It will have method searchRec1(...), searchRec2(...). The searchRec2(...) will count the number of algorithmically essentials compares.

In the main of BinSearch do exactly the following:

1. Do whatever you need to become operational.
2. Construct a Scanner scan.
3. Using the scan read file data.txt that will have less or equal than 20 lines of data. It will reside in the same directory as the compiled code.
4. The file will contain one int per line.
5. Then print on single line the array with 2 blanks between numbers.
6. Then the program will ask the user to give int called val to search for.
7. Then call the searchRec1(...) to search for the val.
8. Print the index in the array where the val is, or if it is not present print -1.
9. It will print the number of compares.

The user will play nice so no need to check for bad data.

Hint:
Look at help2500 for programs using scanner to read file or you can just scavenge your hwk2 which contains similar code for this.

data.txt will look like this (sorted):
22
35
47
66

Problem 2:

Write program BubbleSort. Default package.

This program will read file data2.txt which will reside in the same directory as the compiled code.

1. The program will have class BubbleSort.
2. It will have object scope array of ints called arr of size 20.
3. It will have method bubble(...). The method will use iterative bubblesort and sort the array in place. The bubble(...) will count the number of algorithmically essentials compares.

In the main of BubbleSort do exactly the following:

1. Do whatever you need to become operational.
2. Construct a Scanner scan.
3. Using the scan read file data2.txt that will have less or equal than 20 lines of data. It will reside in the same directory as the compiled code.
4. The file will contain one int per line.
5. Then print on single line the unsorted array with 2 blanks between numbers.
6. Then call the bubble(...) to sort the array.
7. Then print on single line the sorted array with 2 blanks between each number.
8. It will print the number of compares.

The user will play nice so no need to check for bad data.

Hint:
Look at help2500 for programs using scanner to read file or you can just scavenge your hwk2 which contains similar code for this.

data2.txt will look like this:
39
44
2
11
56