Data Structures	Assignment 0 (Part B)
CCO 2103-01, Spring 2024	Due at 5:00pm, Friday, March 22

Hand in your solutions electronically using LearnUs.

Assignment 0 has two parts. The primary purpose of Assignment 0 is to help you acquire basic knowledge of Java programming. For this reason, Assignment 0 will count as 1/10 of an assignment when the final grade is calculated. The course staff will be exceptionally generous in giving out hints, and the grading will also be generous. (This is why you do not want to plagiarize others' work in spite of the fact that Assignment 0 is very similar to what was given in the previous iterations of this course – it is simply not worth it.)

This assignment is *not* a completion points assignment: incorrect solutions will not receive full points.

Submit your source code(s), zipped as *yourStudentID*.zip. For example, if your student ID is 2023000000, then you must zip all your source code(s) into 2023000000.zip and submit this file. Each class should have its own .java file, of which the filename is the same as the class name. Do *not* include your student ID as part of the class names.

This part of the assignment consists of one programming task.

(1) (6 points) Write a program that performs binary search, but by querying an external "oracle." Your program must create a single instance of class Oracle, and communicate with it in order to obtain information about the input. Once the binary search is completed, your program has to report the answer back to the oracle.

The methods of class Oracle are as follows:

- public Oracle()
  Constructs an instance.
- public int getWhatToFind()
  Returns the value that you must find.
- public int getN()
  Returns the number of entries.
- public int getElementAt(int index)
  Returns the value of an entry. The index argument is the (zero-based) index of the entry being queried.
- public void reportAnswer(boolean found, int where)
  Reports the answer. Set the found argument to true if you found the value, and pass the index of the found value as the where argument. Set found to false if you determined that the value does not exist in the input; in this case, the where argument will be ignored. You cannot call this method more than once.

You can assume that the entries are sorted in nondecreasing order. The number of entries is nonnegative.

The entry point of your program must be AsB.main(). Submit your source code(s).

We provide you with a sample oracle to ease your testing. This sample oracle will read its input from input.txt, whose first line specifies the value to find, second line the number of entries n, and the following n lines each of the entry. Upon the report of the answer, the sample oracle will output (to stdout) if your answer is correct and how many queries (i.e., calls to getElementAt()) have been made. When we grade your program, however, we will use a different oracle – this means, for example, that you cannot cheat by trying to directly access the input or stdout.

The following is a sample solution to this assignment, except that it runs linear search. Remember that your program must perform binary search. You can get full points only if your program makes no more than $2\lceil \log_2(n+1) \rceil + 5$ queries. (Note that using the algorithm given in class would result in approximately $\lceil \log_2(n+1) \rceil + 1$ queries.)

```
class AsB {
  public static void main(String[] args) {
    Oracle    o = new Oracle();
    int       n, tofind, a;
    boolean   found;

    n = o.getN();
    tofind = o.getWhatToFind();

    found = false;
    for (int i = 0; i < n; i++) {
      a = o.getElementAt(i);
      if (a == tofind) {
        o.reportAnswer(true, i);
        found = true;
        break;
      }
    }

    if (!found) o.reportAnswer(false, 0);
  }
}
```