University of the Western Cape
Department of Computer Science

CSC212: Assignment 1, Term 4 Due Date: Mon 25 Sep, 11:55pm

IMPORTANT: SUBMISSION IS VIA IKAMVA.

You must submit your Java Code, the output of using both Algorithm 1 and Algorithm 2, the actual running time of both Algorithms, and a report with the answers to Questions d and e.

Suppose you have been given an array *A[1.. n]* of *n* distinct positive and negative integers, sorted in increasing order so that *A[1] < A[2]< …< A[n]*. Your task is to determine if there is an element *A[i] = x* for which both *x* and *-x* are in the array.

Two approaches to solve the problem are presented below:

Algorithm 1:

```
found = false;
for each positive element x in the array
   Perform a sequential search to find -x
         if -x is in the array, print "x and -x are in the array";
```

*Note: A **sequential search** is a linear method that steps through an array until the item it seeks is found.*

Algorithm 2:

```
i = index of first element in array;
j = index of last element;
while (i != j) do
   if the sum of a[i] and a[j] is equal to 0
   then print "a[j] and -a[j] are in the array";
   else
      if the sum of a[i] and a[j] is greater than 0,
        then j=j-1;
        else i=i+1;
```

a. Write Java functions that implement Algorithm 1 and Algorithm 2.

b. Write a driver program (main function) in Java that reads the contents of the sorted array from a text (.txt) file, and then calls each of the two methods to find all the values in the array which has its negative counterpart also in the array. **Assume the array does not contain the number 0 and also does not contain any duplicates**.

c. Two files, *Input1.txt* and *Input2.txt*, will be provided as the input for your array. Test your program with the given input files.

   i. The result of your program must show **all** the positive and negative value pairs in the array using Algorithm 1 and Algorithm 2.

      As an example, your output should look like this:

```
2 and -2 are in the array
6 and -6 are in the array
```

   ii. Your program must also print the **actual** run-time (in nano seconds and seconds) for both Algorithm 1 and Algorithm 2.

d. Give the answers of the following questions in a report (Word document):
   1. Compute and explain the runtime for both algorithms in Big O notation.
   2. From your answer in question d1, which algorithm is more efficient?
   3. Does the **actual** run-time obtained for each algorithm after compiling your program (see question c(ii)) support your answer in question 2d?
   4. Add a screenshot for the run time of each algorithm in the report.

e. Modify your code to also show the number of pairs that were found.
   1. Add a screenshot to show this number in the report.