Part A:

1. Write a function called `printByNumChars` to print all given input names in the list, grouped by number of characters. The runtime must be O(nlogn) where n is the number of names. All names with the same number of characters should be printed on the same line, and names with fewer characters should be printed first. The runtime must be O(nlogn) where n is the number of names in the list. You may use following map to store the string to be printed on the same line. 
   Map<Integer, List<String>> map = new ... // Choose the appropriate map class
   Note: Use a single String to store more than one name may slow down the runtime (but a StringBuilder is fine).

2. Write a function called `printDuplicates` that takes a List of input names and print the names that appear more than once from the list with the count of appearance. The expected runtime must be O(n). The name should be printed in the same order as it appears in the list. Don’t call the list’s contains method, because it is too slow.

3. Write a function called `sumZero` that takes an array of integers, then check whether there’re two distinct numbers in the array to sum to zero, if so, return the index of first number in the first pair. If not, return -1. The expected runtime must be O(n) where n is the array length.

4. Write a function called `isBSTree` that checks whether current Binary Tree is a Binary Search Tree. The expected runtime must be O(n) where n is total number of tree nodes.

Extra Credit: Write a function called longestZigZag that returns how many edge in this longest zig-zag path.

   A zig-zag path: 1. From node n, choose either left or right direction. 2. If the node is not null from this direction, there’s one edge, and move to that child node. 3. From this child node, change direction and repeat all the steps. An empty tree’s longestZigZag is 0.

Change the class name as A3SetMapTree follow by your initials.

Write the method for question 4 and extra credit in class BTree.
Part B

1. Consider following pre-order expression, draw the expression tree.
   *1+*23+456

2. Consider following post-order expression, draw the expression tree.
   123*+45-6-*

3. Draw the following math algorithm as expression tree.
   (3-(5+1) + 2) * (2*3+6) * 3
4. What is the minimum number of node in an AVL tree with height of 3, height 5 and height 8? Show the algorithm below and write down the result.

Consider following diagram showing the state of an AVL tree

```
     r
   / | \         
  l  |  u       
 / | |    \     
 f  n  s  w   
/ | \    \     
 e  i  p  z   
/ | \       
 b  h  j   
```

Diagram 1

5. Write the in-order traversal of the tree.

6. List all single lower case letters whose insertion into an AVL Tree represented by Diagram 1 above would require a rebalance of the tree. (Insert a – z in alphabetic order. The Tree will contain all lower case letters when done).

7. Show how the tree in Diagram 1 is changed when the data element $s$ is removed. Show each state of the tree.