Lab exercise

Write a class called BankAccount. The class must have 3 attributes: accountNo(**int**), holderName(**String**) and balance(**double**).

a) Class constructor will have to set these 3 attributes.

b) Create a balanceChange(double amount) method to reduce balance value by given amount.

c) In the Main method:

1. Create 4 objects (a1, a2, a3, a4) from this class and add them all to an ArrayList called myAccounts. List will only accept BankAccount type.
2. Sort objects by holderName in the list. **(Hint: implement Comparable interface)**
3. Reduce the balances by 500 for all the accounts and print sorted objects in the list.

Write a method `switchPairs` that switches the order of values in an `ArrayList` of `Strings` in a pairwise fashion. Your method should switch the order of the first two values, then switch the order of the next two, switch the order of the next two, and so on. For example, if the list initially stores these values: `{"four", "score", "and", "seven", "years", "ago"}` your method should switch the first pair, `"four", "score"`, the second pair, `"and", "seven"`, and the third pair, `"years", "ago"`, to yield this list: `{"score", "four", "seven", "and", "ago", "years"}`

If there are an odd number of values in the list, the final element is not moved. For example, if the original list had been: `{"to", "be", "or", "not", "to", "be", "hamlet"}` It would again switch pairs of values, but the final value, `"hamlet"` would not be moved, yielding this list: `{"be", "to", "not", "or", "be", "to", "hamlet"}`

Write a method `removeBadPairs` that accepts an `ArrayList` of integers and removes any adjacent pair of integers in the list if the left element of the pair is larger than the right element of the pair. Every pair's left element is an even-numbered index in the list, and every pair's right element is an odd index in the list. For example, suppose a variable called `list` stores the following element values:
`[3, 7, 9, 2, 5, 5, 8, 5, 6, 3, 4, 7, 3, 1]`

We can think of this list as a sequence of pairs: (3, 7), (9, 2), (5, 5), (8, 5), (6, 3), (4, 7), (3, 1). The pairs (9, 2), (8, 5), (6, 3), and (3, 1) are "bad" because the left element is larger than the right one, so these pairs should be removed. So the call of removeBadPairs(list); would change the list to store the following element values:
`[3, 7, 5, 5, 4, 7]`

If the list has an odd length, the last element is not part of a pair and is also considered "bad;" it should therefore be removed by your method.
If an empty list is passed in, the list should still be empty at the end of the call. You may assume that the list passed is not `null`. You may not use any other arrays, lists, or other data structures to help you solve this problem, though you can create as many simple variables as you like.