ITCS 209 Object Oriented Programming	Name:	**Lab Score**	**Challenge Bonus**	
	ID:			
	Group:			

Lab04: Arrays and ArrayLists

You are provided with the BankTester class and BankAccount class. Do not modify these classes. Your task is to implement the methods in the Bank class.

Implement getTotalBalance() which

- Takes no parameters
- Return *totalbalance* in double data type

Implement countBalanceAtLeast(double atLeast) which

- Takes 1 parameter which is the balance required to count an account
- Returns *the number of accounts* having at least the given balance

Implement find(int accountNumber) which

- Takes 1 parameter which is the account number to find
- Returns a *BankAccount* corresponding to the accountNumber, or null if there is no such account.

Implement getMax () which

- Takes no parameters
- Returns a *BankAccount* which the highest balance, or null if the bank has no accounts. If there is more than one BankAccount with the max balance, return one of them.

Implement getMin () which

- Takes no parameters
- Returns a *BankAccount* which the smallest balance, or null if the bank has no accounts. If there is more than one BankAccount with the minimum balance, return one of them.

Noted that: You can see the comments in *Bank* class for more information.

Expected Output:

```
Total balance: 265139.90
Account number with the smallest balance: 6900
Account number with the highest balance: 3185
Account number with having balance at least 5000: 26
Balance of matching account: 1338.23
```

Since this lab aims to test specific aspects of Java/OOP, you can only use the following mechanisms available in Java: loops, arrays, List, ArrayList, primitive data types, Double, String, methods, and classes. You may NOT use Math and other computation libraries, except for Math.sqrt() and Math.pow(). All the computation must be implemented from scratch using loops, arrays, and/or Lists. You may not use other container libraries (e.g., Set, Map, etc.) other than ArrayList. You are also NOT allowed to use third-party Java libraries. Your code should be able to compile and run without having to install external jar files.

Challenge Option01

Implement public List<BankAccount> findDuplicate() in Bank that returns the list of bank accounts that are later found duplicated, so the back staff can further investigate fraudulent activities. Your algorithm must run in ***O(N)*** where N is the number of the bank accounts. Your code can only use lists and arrays. Therefore, Set and Map interfaces are not allowed at this point (Don't worry, you will get to use Set and Map soon).

Expected Output:

```
Duplicate accounts: [BankAccount [acctnumber=6900, balance=60.25], BankAccount
[acctnumber=1969, balance=1044.59], BankAccount [acctnumber=6900, balance=71.25]]
```

Challenge Option02

Write a Java program to check palindrome. A palindrome is a word, phrase, number, or other sequences of characters that can be read the same way from backward or forward. For example, "madam", "mom", "abcba", and "123321" are single-word palindrome. "Don't nod.", "Top spot" and "No lemon, no melon" are multiple words palindrome. *(Punctuation and spaces between the words or letter is allowed.)* "Java", and "Array" are NOT palindrome. The longest palindrome in Oxford dictionary is "tattarrattat".

a. You may write a `Class` or simply have everything in the `main(String[] args)` method.
b. Your program has to accept input `String` from users and check whether the input `String` is a palindrome or not. An example output is shown in the box below:

```
Enter a word or phrase to check if it is a palindrome: tattarrattat
The input word "tattarrattat" is a palindrome

Enter a word or phrase to check if it is a palindrome: I love java
The input phrase "I love Java" is not a palindrome
```
