LEBANESE AMERICAN UNIVERSITY
School of Arts and Science
Department of Computer Science and Mathematics

CSC 243: Introduction to Object Oriented Programming

Lab VI

20. Nov. 2014

Problem 1

Given a sequence of numbers, which should be read into an array, you are asked to write a method called *minMax* which prints the minimum and the maximum number found in the sequence.

The *minMax* method is a void method that should be called from your *main* method and you have the option of sending the array of numbers as a parameter or reading the array directly in the method.

```
public static void minMax(int[] numbers)
```

or

```
public static void minMax()
```

The first line of input is an integer representing the number of test cases. Each test case begins with an integer *N* representing the total numbers in a sequence. *N* integers follow representing the numbers in the sequence.

For each test case, print the maximum and the minimum number using the following format:

Max: *m* Min: *n*

Where *m* is the maximum number and *n* is the minimum number.

Sample Input

```
3
5 1 8 5 3 9
6 5 4 8 3 7 12
4 100 -10 108 30
```

Sample Output

```
Max: 9 Min: 1
Max: 12 Min: 3
Max: 108 Min: -10
```

Problem 2

Given an odd length sequence of strings, which should be read into an array, write a method called *firstMiddleLast* which returns the result of concatenating the first, middle, and last string in the sequence. For example, if the sequence is {"Java", "C", "Python", "C++", "Pascal"}, the string returned by *firstMiddleLast would be JavaPythonPascal.*

The *firstMiddleLast* method should be called from your *main* method and you have the option of sending the array of strings as a parameter or reading the array directly in the method.

```
public static String firstMiddleLast(String[] words)
```

or

```
public static String firstMiddleLast()
```

The first line of input is an integer representing the number of test cases. Each test case begins with an integer *N* representing the number of strings in a sequence. *N* strings follow representing the strings in the sequence.

For each test case, print the string returned by *firstMiddleLast*.

Sample Input	**Sample Output**
3	JavaPythonPascal
5 Java C Python C++ Pascal	ObjectOrientedProgramming
3 Object Oriented Programming	Youknownothing
7 You Java spring know web option nothing	

Problem 3

Given a sequence of distinct numbers, which should be read into an array, and an integer *k*, write a method called *findBefore* which checks if *k* is found in the sequence and returns the number directly before *k*. For example, if the sequence is {4, 5, 2, 6, 9} and *k* is 2, *findBefore* will return 5 since it is the number directly before 2.

If *k* is not found in the sequence, the method should return -1. Also, note that if *k* is the first element in the sequence, the number directly before it would be the last element.

The *findBefore* method should be called from your *main* method and you have the option of sending the array of numbers as a parameter or reading the array directly in the method.

```
public static void findBefore(int[] numbers, int k)
```

or

```
public static void findBefore()
```

Your program will be tested on a number of test cases. Each test case begins with two integers *n* and *k* where *n* is the total numbers in the sequence and *k* is the number you're required to find. The next line will have *n* integers which are the numbers in the sequence. Your program should stop if *n = -1.*

For each test case, print the number find directly before *k* or -1 if *k* is not in the sequence.

Sample Input

```
4 3
1 8 3 9
5 9
1 2 3 4 5
5 8
8 9 2 3 5
-1 8
```

Sample Output

```
8
-1
5
```

Problem 4

A production company held auditions for its new musical. Many singers applied and each had to a voice test by singing increasing tones. The production company will accept the singers if the recording of their voice is always increasing. Being a very busy production company, they do not have time to look over the recordings and determine who will be accepted. Therefore, they hired you the super awesome developer to write a program for them that will determine which singers will be accepted and which will not.

The recording of each singer is a sequence of positive integers of length 8, which should be read into an array. You are required to write the method *isAccepted* which checks if the sequence is in increasing order.

The *isAccepted* method is a boolean method that should be called from your *main* method and you have the option of sending the array of numbers as a parameter or reading the array directly in the method.

```
public static boolean isAccepted(int[] numbers)
```

or

```
public static boolean isAccepted ()
```

The first line of input is an integer representing the number of test cases. Each test case consists of 8 integers representing the numbers in the sequence.

For each test case, print "Accepted" if the singer passed the test, else print "Not Accepted".

Sample Input

```
3
1 5 8 9 10 11 26 30
4 8 9 10 12 7 4 9
15 20 25 27 35 40 50 51
```

Sample Output

```
Accepted
Not Accepted
Accepted
```

Problem 5

Jon Snow does not know how to add! His teacher got frustrated with him so she assigned him an extra problem so maybe he will get better at adding. He is given a sequence of numbers and two integers *a* and *b*. He is required to find the sum of the numbers in the sequence from position *a* to position *b*. For example, if the sequence is {3, 5, 7, 2, 9, 10, 15}, *a* = 2, and *b* = 5, the sum would be 5 + 7 + 2 + 9 = 23. The teacher also does not want the problem to be too easy, so she decided that when giving him the numbers *a* and *b*, it is not always that *a > b.*

Being Jon's epic developer friend, he asked you to write a program for him that will solve his problem. He is happy with his life knowing he cannot add and does not want to learn how.

You are required to write the *findSum* method that should be called from your *main* method and it should take three parameters which are the array of numbers, *a*, and *b*. The method should return the sum of numbers between position *a* and *b* (both inclusive) in the array.

```
public static int findSum(int[] numbers, int a, int b)
```

Your program will be tested on a number of test cases. Each test case begins with three integers *n, a,* and *b* where *n* is the total numbers in the sequence, and *a* and *b* are as described above. The next line will have *n* integers which are the numbers in the sequence. Your program should stop if *n = -1.*

For each test case, print the result returned by *findSum*.

Sample Input

```
7 2 5
3 5 7 2 9 10 15
5 4 1
11 8 14 6 9
-1 8 9
```

Sample Output

```
23
39
```

Problem 6 - Bonus

Write a program that reads a sequence of integers into an array then determines if the sequence can be balanced. A sequence is balanced by splitting it at a certain position and checking if the resulting two sequences have equal sums. For example, the sequence {1, 1, 1, 2 , 1} can be balanced if it is split at position 4. We would have {1, 1, 1} and {2, 1} and the sum of both is equal to 3.

Your program will be tested on a number of test cases. Each test case begins with an integer *n* representing the total numbers in the sequence. The next line will have *n* integers which are the numbers in the sequence. Your program should stop if *n = -1.*

For each test case, print "Yes" if the sequence can be balanced, else print "No".

Sample Input

```
5
1 1 1 2 1
4
2 4 1 5
5
2 1 1 1 4
-1
```

Sample Output

```
Yes
Yes
No
```