Tutorial 06 Sorting and ArrayList

Question One: QuickSort

a) Using the `QuickSort` algorithm complete the diagrams below which show both the index and the value of the elements for TWO passes of the `QuickSort` method [assume that the pivot is always the leftmost element].

i)

	0	1	2	3	4	5	6	7
quickS(	21	3	15	20	11	54	61	19)

pivot is

After one pass the array looks like:

0	1	2	3	4	5	6	7

quickS( ) quickS( )

pivot is is

After two passes the array looks like:

0	1	2	3	4	5	6	7

ii)

	0	1	2	3	4	5	6	7	8	9
quickS(	9	3	8	6	55	4	22	7	11	23)

pivot is

After one pass the array looks like:

0	1	2	3	4	5	6	7	8	9

quickS( ) quickS( )

pivot is is

After two passes the array looks like:

0	1	2	3	4	5	6	7	8	9

b) What is the complexity of the QuickSort algorithm when a good pivot is selected? **O( )**
What is the complexity of the QuickSort algorithm when a 'poor' pivot is selected? **O( )**

Question Two: ArrayList

What is the output after the `T06Q2` method is called in the *main* method? (You could run T06Q2.java and get the output, after you figure it out based on paper.)

```
public static void main (String[] args) {
      T06Q2();
}

private static void T06Q2() {
      ArrayList numbers = new ArrayList();
      for (int i=0; i<5; i++){
            numbers.add(new Integer(i+1));
      }
      ArrayList numbers1 = numbers;
      ArrayList numbers2 = new ArrayList();
      for (int i=0; i<numbers.size(); i++){
            numbers2.add(numbers.get(i));
      }
      ArrayList numbers3 = change(numbers1);
      printArrayListElements(numbers);
      printArrayListElements(numbers1);
      printArrayListElements(numbers2);
      printArrayListElements(numbers3);
}
```

```
	private static ArrayList change( ArrayList arraylist) {
		for( int i=0; i<arraylist.size(); i++ ) {
			if ((Integer)arraylist.get(i)%2==0)
				arraylist.set(i, (Integer)arraylist.get(i)+1);
		}
		return arraylist;
	}

	private static void printArrayListElements(ArrayList arraylist) {
		for( int i=0; i<arraylist.size(); i++ ) {
			System.out.print( (Integer)arraylist.get(i)+ " ");
		}
		System.out.println();
	}
}
```

Question Three: Generic ArrayList

A method with some example code using an `ArrayList` is shown below. Check that you understand how to use an `ArrayList` defined by generics in Java 1.5. Complete the output when the `T06Q3()` method is executed? (As Question Two, you could check your result with the running output of T06Q3.java)

```
	private static void T06Q3() {
		List<String> theAList = new ArrayList<String>();
		System.out.println("1. theAList " + theAList);
		theAList.add(new String("computer"));
		System.out.println("2. theAList " + theAList);
		theAList.add(new String("science"));
		System.out.println("3. theAList " + theAList);
		theAList.add(new String("rock"));
		System.out.println("4. theAList " + theAList);
		theAList.add(1,new String("and"));
		System.out.println("5. theAList "+theAList);
		theAList.remove(1);
		if (theAList.contains(new String("rock")))
			System.out.println("6. contains rock ");
		System.out.println("7. theAList " + theAList);
		System.out.println("8. theAList.isEmpty() " +
		   theAList.isEmpty());
		theAList.set(1, new String("so"));
		System.out.println("9. theAList.get(2) " +
		   theAList.get(2));
		Iterator<String> it = theAList.iterator();
		while (it.hasNext()) {
			System.out.println(it.next().toUpperCase());
		}
		theAList.clear();
		System.out.println("10. theAList " + theAList);
	}
```

Answer

Q1

i)

	0	1	2	3	4	5	6	7
quickS(	21	3	15	20	11	54	61	19)

pivot is 21
after one pass 21 becomes the element at position 5 and the array looks like:

	0	1	2	3	4	5	6	7
	19	3	15	20	11	21	61	54

quickS(19 3 15 20 11) quickS(61 54)

pivot is 19 is 61
after pass two 19 becomes the element at position 3 and the array looks like:

0	1	2	3	4	5	6	7
11	3	15	19	20	21	54	61

ii)

	0	1	2	3	4	5	6	7	8	9
quickS(	9	3	8	6	55	4	22	7	11	23)

pivot is 9
after one pass 9 becomes the element at position 5 and the array looks like:

0	1	2	3	4	5	6	7	8	9
7	3	8	6	4	9	22	55	11	23

quickS(7 3 8 6 4) quickS(22 55 11 23)

pivot is 7 is 22
after pass two 7/22 becomes the element at positions 3/7 and the array looks like:

0	1	2	3	4	5	6	7	8	9
4	3	6	7	8	9	11	22	55	23

Good pivot: O(n log n)

Poor pivot: $\mathbf{O(n^2)}$

Q2

1 3 3 5 5
1 3 3 5 5
1 2 3 4 5
1 3 3 5 5

Q3

1. theAList []
2. theAList [computer]
3. theAList [computer, science]
4. theAList [computer, science, rock]
5. theAList [computer, and, science, rock]
6. contains rock
7. theAList [computer, science, rock]
8. theAList.isEmpty() false
9. theAList.get(2) rock

COMPUTER
SO
ROCK

10. theAList []