Question - 1

Implementation

SCORE: **30 points**

(An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, using all the original letters exactly once.)
Given two strings s and t, write a function to determine if t is an anagram of s.

For example,
s = "anagram", t = "nagaram", return true.
s = "rat", t = "car", return false.

Note:
You may assume the string contains only lowercase alphabets.
Hint:
1. There is **O**(n) solution for this question but your algorithm doesn't have to be **O**(n) as long as you can pass the test cases.
2. You may find *toCharArray()* and *charAt()* methods in String Class useful.
3. You may sort the characters in the given Strings to solve this problem.

Question - 2

Polymorphism

SCORE: **5 points**

What are the results of the following code snippet?

```
class A {

  int value = 1;
  int getValue(){
      return value;
  }
  String outPrint(){
      return "Class A";
  }
}

class B extends A{

   int value = 2;
   int getValue(){
       return value;
  }
   String outPrint(){

       return "Class B";
  }

}

class Test{

  public static void main(String[] args){
    A o = new B();
```

```
    System.out.println(o.getValue());
    System.out.println(o.outPrint());

  }


}
```

- 1
- 2
- Class A
- Class B

Question - 3
Sorting

SCORE: **5 points**

As a developer of your company, you are asked to implement sorting method for processing business data.
If the input data are already sorted in ***most cases***, which one would you choose?

- Insertion Sort
- Selection Sort
- Merge Sort

Question - 4
Sorting

SCORE: **5 points**

As a developer of your company, you are asked to implement a sorting method for processing business data.
If the input data are ***random***, which one would you choose?

- Insertion Sort
- Selection Sort
- Merge Sort

Question - 5
Sorting

SCORE: **5 points**

Given an array in reverse order [6, 5, 4, 3, 2, 1], what are the steps in selection sort?

- 1 5 4 3 2 6, 1 2 4 3 5 6, 1 2 3 4 5 6

- 3 5 4 6 2 1, 3 2 4 6 5 1, 3 2 1 6 5 4, 1 2 3 4 5 6
- 5 6 4 3 2 1, 4 5 6 3 2 1, 3 4 5 6 2 1, 2 3 4 5 6 1, 1 2 3 4 5 6
- 6 5 4 3 1 2, 6 5 4 1 2 3, 6 5 1 2 3 4, 6 1 2 3 4 5, 1 2 3 4 5 6