**Yuliang Jin 986381**

**Problem 1**

[Interview Question] Devise an O(n) algorithm to accomplish this task: Given a none-empty string S of length n, S consists some words separated by spaces. We want to reverse every word in S. For example, given S = “we test coders”, your algorithm is going to return a string with every word in S reversed and separated by spaces. So the result for the above example would be “ew tset sredoc”.

public static String reverseByStack(String s) {

StringBuilder stringBuilder = new StringBuilder();

Stack<Character> characters = new Stack<Character>();

char[] chars = s.toCharArray();

for(int i=0;i<chars.length;i++) {

if(' ' != chars[i]) {

characters.push(chars[i]);

} else {

while (!characters.isEmpty()) {

stringBuilder.append(characters.pop());

}

stringBuilder.append(' ');

}

}

while (!characters.isEmpty()) {

stringBuilder.append(characters.pop());

}

return stringBuilder.toString();

}

**Problem 2**

BSTSort. The BST creation process takes O(nlogn)(?), the output process running time is O(n).

**Problem 3**

For each integer n = 1, 2, 3,..., 7, determine whether there exists a red-black tree having exactly n nodes, with all of them black. Fill out the chart below to tabulate the results:

| **Num nodes n** | **Does there exist a red-black tree with n nodes, all of which are black?** |
| --- | --- |
| 1 | Yes |
| 2 | No |
| 3 | Yes |
| 4 | No |
| 5 | No |
| 6 | No |
| 7 | Yes |

**Problem 4**

For each integer n = 1,2,3,..., 7, determine whether there exists a red-black tree having exactly n nodes, where exactly one of the nodes is red. Fill out the chart below to tabulate the results:

| **Num nodes n** | **Does there exist a red-black tree with n nodes, where exactly one of the nodes is red?** |
| --- | --- |
| 1 | No |
| 2 | Yes |
| 3 | No |
| 4 | Yes |
| 5 | Yes |
| 6 | No |
| 7 | No |