Palindrome recursive

```
public static boolean
palindromecheck(String temp) {
boolean result = false;
temp = temp.toLowerCase();
temp = temp.replaceAll(" ",
"");
temp = temp.replaceAll("\" ",
"");
temp = temp.replaceAll("\\?",
"");
temp = temp.replaceAll("!",
"");
temp = temp.replaceAll("\'",
"");
temp = temp.replaceAll("\\.",
"");
temp = temp.replaceAll(",","");
if (temp.length() > 1) {
if (temp.charAt(0) == temp.char-
At(temp.length()-1)) {
palindromecheck(temp.substr-
ing(1, temp.length()-1));
result = true;
}
else {
result = false;
}
}
return result;
}
```

Sorting(Simple one)

```
public static void
sorting(String[] list) {
for (int x = 0;x < list.leng-
th;x++) {
for (int y = x + 1 ;y < list.l-
ength;y++) {
// indexofarray
if (list[x].compareTo(list[y]) >
0) {
```

Sorting(Simple one) (cont)

```
continue;
}
else if (list[x].compareTo(li-
st[y]) == 0) {
continue;
}
else {
String temp = list[x];
list[x] = list[y];
list[y] = temp;
}
}
System.out.print("Pass: "+x);pr-
intlist(list);
}
}
```

File IO (Write)

```
public class write {
public static void main(String[]
args) {
BufferedWriter writer = null;
File file = new File("log.t-
xt");
try {
System.out.println("test");
writer = new BufferedWriter(new
FileWriter(file));
writer.write("Hello World.W-
at???\n");
writer.append("Hello Mars.\n");
}
catch (FileNotFoundException e)
{
e.printStackTrace();
}
catch (IOException e) {
e.printStackTrace();
```

File IO (Write) (cont)

```
}
finally {
try {
if (writer != null) {
writer.close();
}
}
catch (IOException e) {
e.printStackTrace();
}
}
}
}
```

Node recursive

```
public static void
inOrderTraverse(Node root)
{
if (root == null) {
}
else {
//System.out.print(" "+root.id);
//pre order
inOrderTraverse(root.left);
System.out.print(" "+root.id);
// in order
inOrderTraverse(root.right);
//System.out.print(" "+root.id);
// post order
}
}
```

getSnippet

```
public static String
getSnippet(Word root, int
window)
{
if(root == null) return null;
StringBuilder str = new String-
Builder();
str.append("["+root+"] ");
Word pointer = root.getPreviou-
sWord();
for(int i = 0; i < window; i++)
{ if(pointer == null)
{
break;
}
else
{ str.insert(0, pointer.getWord-
()+" ");
pointer = pointer.getPreviousWo-
rd();
}
}
if(pointer != null) str.in-
sert(0, "... ");

pointer = root.getNextWord();
for(int i = 0; i < window; i++)
{ if(pointer == null)
{
str.append("... ");
break;
}
else
{ str.append(pointer.getWo-
rd()+" ");
pointer = pointer.getNextWord();
}
}
```

getSnippet (cont)

```
if(pointer != null) str.appen-
d("... ");

return str.toString().trim();
}
```

How to create array of int and double

```
int a[]=new int[5];
data = new Double[10];
```

Switch case

```
int day = 4;
switch (day) {
  case 6:
    System.out.println("Today is
Saturday");
    break;
  case 7:
    System.out.println("Today is
Sunday");
    break;
  default:
    System.out.println("Looking
forward to the Weekend");
}
// Outputs "Looking forward to
the Weekend"
```

Compareto

```
result will be int
> 0 : greater
=0 : similar
<0: least
```

isFullBinTree (Node)

```
public static boolean
isFullBinTree(Node root)
{
boolean result = false;
if (root == null) {
result = true;
}
else {
if (root.left != null &&
root.right != null) {
if (!isFullBinTree(root.left) ||
!isFullBinTree(root.right)) {
result = false;
}
else {
result = true;
}
}
else if (root.left == null &&
root.right == null) {
result = true;
}
else {
result = false;
}
}
return result;
}
```

binaryfind

```
public static boolean
binaryfind(int[][] matrix, int
window) {
    boolean check = false;
    int length = matrix-
[0].length * matrix[1].length;
int low = 0;
int high = sortedWords.size() -
1;
int mid = 0;
while (low <= high) {
mid = low + ((high - low)/2);
if (sortednumber[mid] == target)
{
return true;
}
else if (sortednumber[mid] <
target) {
low = mid + 1;
}
else {
high = mid - 1;
}
}
return check;
}
```

File IO (Read apache)

```
//load file
String text = null;
try {
text = FileUtils.readFileToStri-
ng(new File(filename), "UTF-8");
} catch (IOException e) {
e.printStackTrace();
}
```

File IO (Read apache) (cont)

```
//clean text
text = text.toLowerCase().repla-
ceAll("\\W+", " ").replaceAll-
("\\s\\w\\s", " ");

//System.out.println(text);
//tokenize words
String[] tokens = text.split("-
\\s+");

//store tokens in words
sortedWords = new ArrayList<Wo-
rd>();
for(int i = 0; i < tokens.le-
ngth; i++)
{ sortedWords.add(new Word(toke-
ns[i],i));
if(i > 0)
{ sortedWords.get(i).setPrevi-
ousWord(sortedWords.get(i-1));
sortedWords.get(i-1).setNextWo-
rd(sortedWords.get(i));
}
}
//sort word
Collections.sort(sortedWords);
```