Recursion - Take Home Problems Solutions

Solutions

1.

```
function isPalindromeRecursive(word){
return isPalindromeHelper (word, 0, word. length-1);
}
function isPalindromeHelper (word, beginPos, endPos) {
if (beginPos >= endPos) {
return true;
}
if (word. charAt (beginPos) != word. charAt (endPos)) {
return false;
}
else {
return isPalindromeHelper(word, beginPos + 1, endPos - 1);
}
}
```

2.

```
function base10ToString(n) {
var binaryString = "";
function base10ToStringHelper (n) {
if (n < 2) {
binaryString += n;
return;
} else {
base10ToStringHelper (Math.floor (n / 2));
base10ToStringHelper (n % 2);
}
}
base10ToStringHelper(n);
return binaryString;
}
console.log(base10ToString(15)); // 1111
```

3.

```
function swap (strArr, index1, index2) {
var temp = strArr[index1];
strArr[index1] = strArr[index2];
strArr[index2] = temp;
}
function permute (strArr, begin, end) {
if (begin == end) {
console.log(strArr);
} else {
```

```
for (var i = begin; i < end + 1; i++) {
swap (strArr, begin, i);
permute (strArr, begin + 1, end);
swap (strArr, begin, i);
}
}
}
function permuteArray(strArr){
permute (strArr, 0, strArr.length - 1);
}

permuteArray(["A", "B", "C"]);
// ["A", "B", "C"1
// ["A", "C", "B"]
// ["B", "A", "C"1
// ["B", "C", "A"]
// ["C", "B", "A"]
// ["C", "A", "B"]
```

4.

```
function countdigits(n,k)
{
    if (n == 0) // Base Case
        return 0;
    // Extracting least significant digit
    var digit = Math.floor(n % 10);
    if (digit === k)
        return 1 + countdigits(n/10, k);

    return countdigits(n/10, k);
}
var n = 86487;
var k = 8;
document.write(countdigits(n, k));
```

5.

```
function printPascal(n)
{
    for(line = 1; line <= n; line++)
    {
     var C=1;// used to represent C(line, i)
     for(i = 1; i <= line; i++){
        // The first value in a line is always 1
        document.write(C+" ");
        C = C * (line - i) / i;
    }
      document.write("<br>");
    }
}
var n = 6;
```

```
printPascal(n);
```