Cover Page

Name: David Chen Salas

Section: 2023 Fall Term (1) Algorithms I CSCI 700 231[25504] (Queens College)

Project#: Project 2

Project Name: Huffman Coding Part 1(Java)

Due Date: 9/21/2023 Thursday before midnight

Algorithm Steps:

```
Step 0: InFile, outFile1, deBugFile  open via args [0], args [1], args [2]
Step 1: computeCharCounts (inFile, charCountAry, deBugFile) // On your own,
    // You may pass deBugFile to write some debugging prints
Step 2: printCountAry (charCountAry, outFile1) // with caption “Below is character counts”
Step 3: LL  creates a LLlist using constructor and assign
    LL.listHead  get a treeNode with (“dummy” , 0, ‘’, null, null, null) // ‘’ is an empty string
Step 4: constructHuffmanLList (LL.listHead, charCountAry, deBugFile)
    Step 5: printList (LL.listHead, outFile1) // with caption “Below is the ordered Huffman ordered Linked list.”
Step 6: constructHuffmanBinTree (LL.listHead, deBugFile)
    Step 7: (binTree) HuffmanTree  create a binTree node using binTree constructor and assign
Step 8: HuffmanTree.Root  LL.listHead.next
Step 9: preOrder (HuffmanTree.Root, outFile1)
    // with caption “Below is preOrder of the Huffman Binary Tree”
    inOrder (HuffmanTree.Root, outFile1) // with caption.
    postOrder (HuffmanTree.Root, outFile1) // with caption.
Step 10: constructCharCode (HuffmanTree.Root, ‘’, codeTable) // ‘’ is an empty string; see algorithm below.in
    Step 11: close all files.
```

Illustration

No need for this project

Source Code

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;
import java.lang.String;

public class Main {

    static Scanner inFile;
    static FileWriter outFile1;
    static FileWriter deBugFile;
    static HuffmanCode huffmanCode;
    static LLlist LL;
    static binTree HuffmanTree;

//    static BufferedReader inFile;

    public static void main(String[] args) throws IOException {

        inFile = new Scanner(new FileReader(args[0]));
        inFile.useDelimiter("");
//        inFile = new BufferedReader(new FileReader("HuffmanCodingData.txt"));
        outFile1 = new FileWriter(args[1]);
        deBugFile = new FileWriter(args[2]);
        huffmanCode = new HuffmanCode();

        computeCharCounts(inFile, huffmanCode.charCountAry, deBugFile);
        printCountAry(huffmanCode.charCountAry, outFile1);
        LL = new LLlist();
        constructHuffmanLList(LL.listHead, huffmanCode.charCountAry, deBugFile);
        printList(LL.listHead, outFile1);
        constructHuffmanBinTree(LL.listHead, deBugFile);
        HuffmanTree = new binTree(LL.listHead.next);

        outFile1.write("\n**Below is preOrder of the Huffman Binary Tree**\n");
        preOrder(HuffmanTree.Root, outFile1);
        outFile1.write("\n**Below is inOrder of the Huffman Binary Tree**\n");
        inOrder(HuffmanTree.Root, outFile1);
        outFile1.write("\n**Below is postOrder of the Huffman Binary Tree**\n");
        postOrder(HuffmanTree.Root, outFile1);
        constructCharCode(HuffmanTree.Root, "", huffmanCode.codeTable);

        inFile.close();
        outFile1.close();
        deBugFile.close();
    }

    public static void printNode(treeNode T, FileWriter file) throws IOException {
        String chStr, leftChr, rightChr, nextChr;

        switch (T.chStr){
            case " ":
                chStr = "\'space\'";
                break;
            case "\r":
                chStr = "\\r";
                break;
            case "\n":
                chStr = "NL";
                break;
            default:
                chStr = T.chStr;
        }

        if(T.left == null) leftChr = "null";
        else {
            switch (T.left.chStr){
                case " ":
                    leftChr = "\'space\'";
                    break;
                case "\r":
                    leftChr = "\\r";
                    break;
                case "\n":
                    leftChr = "NL";
                    break;
                default:
                    leftChr = T.left.chStr;
            }
        }

        if(T.right == null) rightChr = "null";
```

```
        else {
            switch (T.right.chStr){
                case " ":
                    rightChr = "\'space\'";
                    break;
                case "\r":
                    rightChr = "\\r";
                    break;
                case "\n":
                    rightChr = "NL";
                    break;
                default:
                    rightChr = T.right.chStr;
            }
        }

        if(T.next == null) nextChr = "null";
        else {
            switch (T.next.chStr) {
                case " ":
                    nextChr = "\'space\'";
                    break;
                case "\r":
                    nextChr = "\\r";
                    break;
                case "\n":
                    nextChr = "NL";
                    break;
                default:
                    nextChr = T.next.chStr;
            }
        }

        file.write("(");
        file.write(chStr+",  " + T.frequency+",  " + T.code+",  " + leftChr+",  " + rightChr+",  " +
nextChr);
        file.write(")\n");
    }

    public static void printList(treeNode listHead, FileWriter file) throws IOException {
        if(file.toString()=="outFile1"){
            file.write("Below is the ordered Huffman ordered Linked list.\n");
        }
        else {
            file.write("Printing list in constructHuffmanLList method.\n");
        }
        treeNode pNode = listHead;
        while(pNode!=null){
            printNode(pNode, file);
            pNode = pNode.next;
        }
    }

    public static treeNode findSpot(treeNode listHead, treeNode newNode, FileWriter deBugFile)
throws IOException {
        deBugFile.write("Entering findSpot method!\n");
        treeNode spot = listHead;
        while (spot.next != null) {
            deBugFile.write("In findSpot: Spot.next's frequency is " + spot.next.frequency +
                " and newNode's frequency is " + newNode.frequency + "\n");
            if (spot.next.frequency < newNode.frequency) {
                spot = spot.next;
            }
            else{
                break;
            }
        }
        deBugFile.write("Leaving findSpot method!\n");
        return spot;
    }

    public static void insertOneNode(treeNode spot, treeNode newNode){
        newNode.next = spot.next;
        spot.next = newNode;
    }

    public static void preOrder(treeNode rootNode, FileWriter file) throws IOException {
        if(rootNode.left==null && rootNode.right==null){
            printNode(rootNode, file);
        }
        else {
            printNode(rootNode,file);
            preOrder(rootNode.left, file);
            preOrder(rootNode.right, file);
        }
    }

    public static void inOrder(treeNode rootNode, FileWriter file) throws IOException{

        if(rootNode.left==null && rootNode.right==null){
            printNode(rootNode, file);
        }
        else {
            inOrder(rootNode.left, file);
            printNode(rootNode,file);
            inOrder(rootNode.right,file);
        }
    }

    public static void postOrder(treeNode rootNode, FileWriter file) throws IOException {

        if(rootNode.left==null && rootNode.right==null){
            printNode(rootNode, file);
        }
        else {
            postOrder(rootNode.left, file);
            postOrder(rootNode.right, file);
            printNode(rootNode,file);
        }
    }

    public static boolean isLeaf(treeNode node){
        if(node.left==null && node.right==null){
            return true;
        }
        return false;
    }

    public static void computeCharCounts(Scanner inFile, int[] charCountAry, FileWriter
deBugFile) throws IOException {
        deBugFile.write("Entering computeCharCounts method!\n");
        char c;
        while(inFile.hasNext()) {
            c = (char)inFile.next().charAt(0);
            charCountAry[(int)c]++;
            if(c=='\r' || c==' ' || c=='\n'){
                deBugFile.write("character " + (int)c + " read\n");
            }
            else {
                deBugFile.write("character " + c + " read\n");
            }
        }
        deBugFile.write("Leaving computeCharCounts method!\n");
    }

    public static void printCountAry(int[] charCountAry, FileWriter outFile1) throws IOException
{
        outFile1.write("**Below is character counts**\n");
        outFile1.write("Index\tChar\tCount\n");
        outFile1.write("=====================\n");
        for(int i=0; i<charCountAry.length; i++){
            if(charCountAry[i]!=0) {
                if(i==10){
                    outFile1.write(i + "\tNL\t" + charCountAry[i] + "\n");
                }
                else if(i==13){
                    outFile1.write(i + "\t\\r\t" + charCountAry[i] + "\n");
                }
                else if(i==32){
                    outFile1.write(i + "\t'space'\t" + charCountAry[i] + "\n");
                }
                else{
                    outFile1.write(i + "\t" + (char)i + "\t" + charCountAry[i] + "\n");
                }
            }
        }
    }

    public static void constructHuffmanLList(treeNode listHead, int[] charCountAry, FileWriter
deBugFile) throws IOException {
        deBugFile.write("Entering constructHuffmanLList method!\n");
        char chr;
        int frequency;
        for(int i=0; i<256; i++){
            if(charCountAry[i]>0){
                chr = (char)i;
                frequency = charCountAry[i];
                treeNode newNode = new treeNode(""+chr, frequency,"",null, null, null);
                printNode(newNode, deBugFile);
                treeNode spot = findSpot(listHead, newNode, deBugFile);
                insertOneNode(spot, newNode);
                printList(listHead, deBugFile);
            }
        }
        deBugFile.write("Leaving constructHuffmanLList method!\n");
    }

    public static void constructHuffmanBinTree(treeNode listHead, FileWriter deBugFile) throws
IOException{
        deBugFile.write("Entering constructHuffmanBinTree method!\n");
        while(listHead.next.next!=null) {
            treeNode leftNode = listHead.next;
            treeNode rightNode = listHead.next.next;
            String cStr = leftNode.chStr + rightNode.chStr;
            StringBuilder s = new StringBuilder();
            for(char x: cStr.toCharArray()){
                if(x == ' ') s.append("\'space\'");
                else if(x == '\r') s.append("\\r");
                else if(x == '\n') s.append("NL");
                else s.append(x);
            }
            int frequency = leftNode.frequency + rightNode.frequency;
            treeNode newNode = new treeNode(s.toString(), frequency, "", leftNode, rightNode,
null);
            printNode(newNode, deBugFile);
            treeNode spot = findSpot(listHead, newNode, deBugFile);
            insertOneNode(spot, newNode);
            listHead.next = listHead.next.next.next;
            printList(listHead, deBugFile);
        }
        deBugFile.write("Leaving constructHuffmanBinTree method!\n");
    }

    public static void constructCharCode(treeNode T, String code, String[] codeTable){
        if(isLeaf(T)){
            T.code = code;
            codeTable[(int)T.chStr.charAt(0)] = code;
        }
        else{
            constructCharCode(T.left, code+"0", codeTable);
            constructCharCode(T.right, code+"1", codeTable);
        }
    }

}

class treeNode{
    String chStr;
    int frequency;
    String code;
    treeNode left;
    treeNode right;
    treeNode next;

    treeNode(String chStr, int frequency, String code, treeNode left,
             treeNode right, treeNode next){
        this.chStr = chStr;
        this.frequency = frequency;
        this.code = code;
        this.left = left;
        this.right = right;
        this.next = next;
    }
}

class LLlist{
    treeNode listHead;
    LLlist(){
        listHead = new treeNode("dummy", 0, "", null, null, null);
    }
}

class binTree{
    treeNode Root;
    binTree(treeNode Root){
        this.Root = Root;
    }
}

class HuffmanCode{
    int[] charCountAry;
    String[] codeTable;

    HuffmanCode(){
        charCountAry = new int[256];
        codeTable = new String[256];
    }
}
```

Program Output

Below is outFile1:

```
**Below is character counts**
Index   Char    Count
====================
10      NL      399
13      \r      399
32      'space' 4253
```

```
39      '       15
40      (       1
41      )       1
44      ,       373
45      -       1
46      .       181
49      1       23
51      3       11
52      4       1
53      5       2
54      6       11
56      8       13
57      9       11
59      ;       1
65      A       47
66      B       23
67      C       34
68      D       1
70      F       12
71      G       46
72      H       3
73      I       44
74      J       1
76      L       22
77      M       6
78      N       33
79      O       6
80      P       23
83      S       40
84      T       49
85      U       25
87      W       34
97      a       1627
98      b       271
99      c       456
100     d       876
101     e       2547
102     f       478
103     g       406
104     h       1240
105     i       1131
106     j       1
107     k       81
108     l       662
109     m       269
110     n       1321
111     o       1436
112     p       216
113     q       11
114     r       1226
115     s       862
116     t       1914
117     u       343
118     v       323
119     w       338
120     x       6
121     y       260
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  j)
(j,  1,  ,  null,  null,  J)
(J,  1,  ,  null,  null,  D)
(D,  1,  ,  null,  null,  ;)
```

```
(;,  1,  ,  null,  null,  4)
(4,  1,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  5)
(5,  2,  ,  null,  null,  H)
(H,  3,  ,  null,  null,  x)
(x,  6,  ,  null,  null,  O)
(O,  6,  ,  null,  null,  M)
(M,  6,  ,  null,  null,  q)
(q,  11,  ,  null,  null,  9)
(9,  11,  ,  null,  null,  6)
(6,  11,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  F)
(F,  12,  ,  null,  null,  8)
(8,  13,  ,  null,  null,  ')
(',  15,  ,  null,  null,  L)
(L,  22,  ,  null,  null,  P)
(P,  23,  ,  null,  null,  B)
(B,  23,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  U)
(U,  25,  ,  null,  null,  N)
(N,  33,  ,  null,  null,  W)
(W,  34,  ,  null,  null,  C)
(C,  34,  ,  null,  null,  S)
(S,  40,  ,  null,  null,  I)
(I,  44,  ,  null,  null,  G)
(G,  46,  ,  null,  null,  A)
(A,  47,  ,  null,  null,  T)
(T,  49,  ,  null,  null,  k)
(k,  81,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  p)
(p,  216,  ,  null,  null,  y)
(y,  260,  ,  null,  null,  m)
(m,  269,  ,  null,  null,  b)
(b,  271,  ,  null,  null,  v)
(v,  323,  ,  null,  null,  w)
(w,  338,  ,  null,  null,  u)
(u,  343,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  g)
(g,  406,  ,  null,  null,  c)
(c,  456,  ,  null,  null,  f)
(f,  478,  ,  null,  null,  l)
(l,  662,  ,  null,  null,  s)
(s,  862,  ,  null,  null,  d)
(d,  876,  ,  null,  null,  i)
(i,  1131,  ,  null,  null,  r)
(r,  1226,  ,  null,  null,  h)
(h,  1240,  ,  null,  null,  n)
(n,  1321,  ,  null,  null,  o)
(o,  1436,  ,  null,  null,  a)
(a,  1627,  ,  null,  null,  t)
(t,  1914,  ,  null,  null,  e)
(e,  2547,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)

**Below is preOrder of the Huffman Binary Tree**
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space',  24446,  ,
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.,  o,\rNLG1OMFATpagcsdfymt'space',  null)
```

```
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB., 10033, , ibU8'NWCSkvwrh, enluD;jJ)(4-5Hx63q9LIPB.,
o,\rNLG1OMFATpagcsdfymt'space')
(ibU8'NWCSkvwrh, 4804, , ibU8'NWCSkvw, rh, enluD;jJ)(4-5Hx63q9LIPB.)
(ibU8'NWCSkvw, 2338, , i, bU8'NWCSkvw, rh)
(i, 1131, , null, null, bU8'NWCSkvw)
(bU8'NWCSkvw, 1207, , bU8'NWCSk, vw, r)
(bU8'NWCSk, 546, , b, U8'NWCSk, vw)
(b, 271, , null, null, U8'NWCSk)
(U8'NWCSk, 275, , U8'NW, CSk, v)
(U8'NW, 120, , U8', NW, CSk)
(U8', 53, , U, 8', NW)
(U, 25, , null, null, 8')
(8', 28, , 8, ', N)
(8, 13, , null, null, ')
(', 15, , null, null, D;jJ)(4-5Hx)
(NW, 67, , N, W, CS)
(N, 33, , null, null, W)
(W, 34, , null, null, C)
(CSk, 155, , CS, k, D;jJ)(4-5Hx63q9LIPB)
(CS, 74, , C, S, k)
(C, 34, , null, null, S)
(S, 40, , null, null, D;jJ)(4-5Hx63)
(k, 81, , null, null, D;jJ)(4-5Hx63q9L)
(vw, 661, , v, w, l)
(v, 323, , null, null, w)
(w, 338, , null, null, u)
(rh, 2466, , r, h, e)
(r, 1226, , null, null, h)
(h, 1240, , null, null, n)
(enluD;jJ)(4-5Hx63q9LIPB., 5229, , e, nluD;jJ)(4-5Hx63q9LIPB., o,\rNLG1OMFATpagcs)
(e, 2547, , null, null, nluD;jJ)(4-5Hx63q9LIPB.)
(nluD;jJ)(4-5Hx63q9LIPB., 2682, , n, luD;jJ)(4-5Hx63q9LIPB., o,\rNLG1OMFATp)
(n, 1321, , null, null, luD;jJ)(4-5Hx63q9LIPB.)
(luD;jJ)(4-5Hx63q9LIPB., 1361, , l, uD;jJ)(4-5Hx63q9LIPB., o)
(l, 662, , null, null, uD;jJ)(4-5Hx63q9LIPB.)
(uD;jJ)(4-5Hx63q9LIPB., 699, , u, D;jJ)(4-5Hx63q9LIPB., ,\r)
(u, 343, , null, null, D;jJ)(4-5Hx63q9LIPB.)
(D;jJ)(4-5Hx63q9LIPB., 356, , D;jJ)(4-5Hx63q9LIPB, ., ,)
(D;jJ)(4-5Hx63q9LIPB, 175, , D;jJ)(4-5Hx63q9L, IPB, .)
(D;jJ)(4-5Hx63q9L, 85, , D;jJ)(4-5Hx63, q9L, IPB)
(D;jJ)(4-5Hx63, 41, , D;jJ)(4-5Hx, 63, q9L)
(D;jJ)(4-5Hx, 19, , D;jJ)(4-, 5Hx, 63)
(D;jJ)(4-, 8, , D;jJ, )(4-, 5Hx)
(D;jJ, 4, , D;, jJ, )(4-)
(D;, 2, , D, ;, jJ)
(D, 1, , null, null, ;)
(;, 1, , null, null, 4)
(jJ, 2, , j, J, 5)
(j, 1, , null, null, J)
(J, 1, , null, null, D)
()(4-, 4, , )(, 4-, 5H)
()(, 2, , ), (, 4-)
(), 1, , null, null, ()
((, 1, , null, null, )()
(4-, 2, , 4, -, D;)
(4, 1, , null, null, -)
(-, 1, , null, null, ))
(5Hx, 11, , 5H, x, q)
(5H, 5, , 5, H, x)
(5, 2, , null, null, H)
(H, 3, , null, null, D;jJ)
(x, 6, , null, null, O)
```

```
(63,  22,  ,  6,  3,  q9)
(6,  11,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  OM)
(q9L,  44,  ,  q9,  L,  I)
(q9,  22,  ,  q,  9,  L)
(q,  11,  ,  null,  null,  9)
(9,  11,  ,  null,  null,  6)
(L,  22,  ,  null,  null,  P)
(IPB,  90,  ,  I,  PB,  G1OMF)
(I,  44,  ,  null,  null,  PB)
(PB,  46,  ,  P,  B,  G)
(P,  23,  ,  null,  null,  B)
(B,  23,  ,  null,  null,  1)
(.,  181,  ,  null,  null,  G1OMFAT)
(o,\rNLG1OMFATpagcsdfymt'space',  14413,  ,  o,\rNLG1OMFATpagcs,  dfymt'space',
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space')
(o,\rNLG1OMFATpagcs,  6363,  ,  o,\rNLG1OMFATp,  agcs,  dfymt'space')
(o,\rNLG1OMFATp,  3012,  ,  o,  ,\rNLG1OMFATp,  agcs)
(o,  1436,  ,  null,  null,  ,\rNLG1OMFATp)
(,\rNLG1OMFATp,  1576,  ,  ,\r,  NLG1OMFATp,  a)
(,\r,  772,  ,  ,,  \r,  NLG1OMFATp)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NLG1OMFATp,  804,  ,  NL,  G1OMFATp,  gc)
(NL,  399,  ,  null,  null,  G1OMFATp)
(G1OMFATp,  405,  ,  G1OMFAT,  p,  g)
(G1OMFAT,  189,  ,  G1OMF,  AT,  p)
(G1OMF,  93,  ,  G,  1OMF,  AT)
(G,  46,  ,  null,  null,  1OMF)
(1OMF,  47,  ,  1,  OMF,  A)
(1,  23,  ,  null,  null,  OMF)
(OMF,  24,  ,  OM,  F,  U)
(OM,  12,  ,  O,  M,  F)
(O,  6,  ,  null,  null,  M)
(M,  6,  ,  null,  null,  D;jJ)(4-)
(F,  12,  ,  null,  null,  8)
(AT,  96,  ,  A,  T,  U8'NW)
(A,  47,  ,  null,  null,  T)
(T,  49,  ,  null,  null,  U8')
(p,  216,  ,  null,  null,  y)
(agcs,  3351,  ,  a,  gcs,  dfymt)
(a,  1627,  ,  null,  null,  gcs)
(gcs,  1724,  ,  gc,  s,  dfym)
(gc,  862,  ,  g,  c,  s)
(g,  406,  ,  null,  null,  c)
(c,  456,  ,  null,  null,  f)
(s,  862,  ,  null,  null,  d)
(dfymt'space',  8050,  ,  dfymt,  'space',  ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.)
(dfymt,  3797,  ,  dfym,  t,  'space')
(dfym,  1883,  ,  d,  fym,  t)
(d,  876,  ,  null,  null,  fym)
(fym,  1007,  ,  f,  ym,  i)
(f,  478,  ,  null,  null,  ym)
(ym,  529,  ,  y,  m,  bU8'NWCSk)
(y,  260,  ,  null,  null,  m)
(m,  269,  ,  null,  null,  b)
(t,  1914,  ,  null,  null,  ibU8'NWCSkvw)
('space',  4253,  ,  null,  null,  ibU8'NWCSkvwrh)

**Below is inOrder of the Huffman Binary Tree**
(i,  1131,  ,  null,  null,  bU8'NWCSkvw)
(ibU8'NWCSkvw,  2338,  ,  i,  bU8'NWCSkvw,  rh)
```

```
(b, 271, , null, null, U8'NWCSk)
(bU8'NWCSk, 546, , b, U8'NWCSk, vw)
(U, 25, , null, null, 8')
(U8', 53, , U, 8', NW)
(8, 13, , null, null, ')
(8', 28, , 8, ', N)
(', 15, , null, null, D;jJ)(4-5Hx)
(U8'NW, 120, , U8', NW, CSk)
(N, 33, , null, null, W)
(NW, 67, , N, W, CS)
(W, 34, , null, null, C)
(U8'NWCSk, 275, , U8'NW, CSk, v)
(C, 34, , null, null, S)
(CS, 74, , C, S, k)
(S, 40, , null, null, D;jJ)(4-5Hx63)
(CSk, 155, , CS, k, D;jJ)(4-5Hx63q9LIPB)
(k, 81, , null, null, D;jJ)(4-5Hx63q9L)
(bU8'NWCSkvw, 1207, , bU8'NWCSk, vw, r)
(v, 323, , null, null, w)
(vw, 661, , v, w, l)
(w, 338, , null, null, u)
(ibU8'NWCSkvwrh, 4804, , ibU8'NWCSkvw, rh, enluD;jJ)(4-5Hx63q9LIPB.)
(r, 1226, , null, null, h)
(rh, 2466, , r, h, e)
(h, 1240, , null, null, n)
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB., 10033, , ibU8'NWCSkvwrh, enluD;jJ)(4-5Hx63q9LIPB.,
o,\rNLG1OMFATpagcsdfymt'space')
(e, 2547, , null, null, nluD;jJ)(4-5Hx63q9LIPB.)
(enluD;jJ)(4-5Hx63q9LIPB., 5229, , e, nluD;jJ)(4-5Hx63q9LIPB., o,\rNLG1OMFATpagcs)
(n, 1321, , null, null, luD;jJ)(4-5Hx63q9LIPB.)
(nluD;jJ)(4-5Hx63q9LIPB., 2682, , n, luD;jJ)(4-5Hx63q9LIPB., o,\rNLG1OMFATp)
(l, 662, , null, null, uD;jJ)(4-5Hx63q9LIPB.)
(luD;jJ)(4-5Hx63q9LIPB., 1361, , l, uD;jJ)(4-5Hx63q9LIPB., o)
(u, 343, , null, null, D;jJ)(4-5Hx63q9LIPB.)
(uD;jJ)(4-5Hx63q9LIPB., 699, , u, D;jJ)(4-5Hx63q9LIPB., ,\r)
(D, 1, , null, null, ;)
(D;, 2, , D, ;, jJ)
(;, 1, , null, null, 4)
(D;jJ, 4, , D;, jJ, )(4-)
(j, 1, , null, null, J)
(jJ, 2, , j, J, 5)
(J, 1, , null, null, D)
(D;jJ)(4-, 8, , D;jJ, )(4-, 5Hx)
(), 1, , null, null, ()
()(, 2, , ), (, 4-)
((, 1, , null, null, )()
()(4-, 4, , )(, 4-, 5H)
(4, 1, , null, null, -)
(4-, 2, , 4, -, D;)
(-, 1, , null, null, ))
(D;jJ)(4-5Hx, 19, , D;jJ)(4-, 5Hx, 63)
(5, 2, , null, null, H)
(5H, 5, , 5, H, x)
(H, 3, , null, null, D;jJ)
(5Hx, 11, , 5H, x, q)
(x, 6, , null, null, O)
(D;jJ)(4-5Hx63, 41, , D;jJ)(4-5Hx, 63, q9L)
(6, 11, , null, null, 3)
(63, 22, , 6, 3, q9)
(3, 11, , null, null, OM)
(D;jJ)(4-5Hx63q9L, 85, , D;jJ)(4-5Hx63, q9L, IPB)
(q, 11, , null, null, 9)
```

```
(q9,  22,  ,  q,  9,  L)
(9,  11,  ,  null,  null,  6)
(q9L,  44,  ,  q9,  L,  I)
(L,  22,  ,  null,  null,  P)
(D;jJ)(4-5Hx63q9LIPB,  175,  ,  D;jJ)(4-5Hx63q9L,  IPB,  .)
(I,  44,  ,  null,  null,  PB)
(IPB,  90,  ,  I,  PB,  G1OMF)
(P,  23,  ,  null,  null,  B)
(PB,  46,  ,  P,  B,  G)
(B,  23,  ,  null,  null,  1)
(D;jJ)(4-5Hx63q9LIPB.,  356,  ,  D;jJ)(4-5Hx63q9LIPB,  .,  ,)
(.,  181,  ,  null,  null,  G1OMFAT)
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space',  24446,  ,
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.,  o,\rNLG1OMFATpagcsdfymt'space',  null)
(o,  1436,  ,  null,  null,  ,\rNLG1OMFATp)
(o,\rNLG1OMFATp,  3012,  ,  o,  ,\rNLG1OMFATp,  agcs)
(,,  373,  ,  null,  null,  \r)
(,\r,  772,  ,  ,,  \r,  NLG1OMFATp)
(\r,  399,  ,  null,  null,  NL)
(,\rNLG1OMFATp,  1576,  ,  ,\r,  NLG1OMFATp,  a)
(NL,  399,  ,  null,  null,  G1OMFATp)
(NLG1OMFATp,  804,  ,  NL,  G1OMFATp,  gc)
(G,  46,  ,  null,  null,  1OMF)
(G1OMF,  93,  ,  G,  1OMF,  AT)
(1,  23,  ,  null,  null,  OMF)
(1OMF,  47,  ,  1,  OMF,  A)
(O,  6,  ,  null,  null,  M)
(OM,  12,  ,  O,  M,  F)
(M,  6,  ,  null,  null,  D;jJ)(4-)
(OMF,  24,  ,  OM,  F,  U)
(F,  12,  ,  null,  null,  8)
(G1OMFAT,  189,  ,  G1OMF,  AT,  p)
(A,  47,  ,  null,  null,  T)
(AT,  96,  ,  A,  T,  U8'NW)
(T,  49,  ,  null,  null,  U8')
(G1OMFATp,  405,  ,  G1OMFAT,  p,  g)
(p,  216,  ,  null,  null,  y)
(o,\rNLG1OMFATpagcs,  6363,  ,  o,\rNLG1OMFATp,  agcs,  dfymt'space')
(a,  1627,  ,  null,  null,  gcs)
(agcs,  3351,  ,  a,  gcs,  dfymt)
(g,  406,  ,  null,  null,  c)
(gc,  862,  ,  g,  c,  s)
(c,  456,  ,  null,  null,  f)
(gcs,  1724,  ,  gc,  s,  dfym)
(s,  862,  ,  null,  null,  d)
(o,\rNLG1OMFATpagcsdfymt'space',  14413,  ,  o,\rNLG1OMFATpagcs,  dfymt'space',
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space')
(d,  876,  ,  null,  null,  fym)
(dfym,  1883,  ,  d,  fym,  t)
(f,  478,  ,  null,  null,  ym)
(fym,  1007,  ,  f,  ym,  i)
(y,  260,  ,  null,  null,  m)
(ym,  529,  ,  y,  m,  bU8'NWCSk)
(m,  269,  ,  null,  null,  b)
(dfymt,  3797,  ,  dfym,  t,  'space')
(t,  1914,  ,  null,  null,  ibU8'NWCSkvw)
(dfymt'space',  8050,  ,  dfymt,  'space',  ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.)
('space',  4253,  ,  null,  null,  ibU8'NWCSkvwrh)

**Below is postOrder of the Huffman Binary Tree**
(i,  1131,  ,  null,  null,  bU8'NWCSkvw)
(b,  271,  ,  null,  null,  U8'NWCSk)
```

```
(U, 25, , null, null, 8')
(8, 13, , null, null, ')
(', 15, , null, null, D;jJ)(4-5Hx)
(8', 28, , 8, ', N)
(U8', 53, , U, 8', NW)
(N, 33, , null, null, W)
(W, 34, , null, null, C)
(NW, 67, , N, W, CS)
(U8'NW, 120, , U8', NW, CSk)
(C, 34, , null, null, S)
(S, 40, , null, null, D;jJ)(4-5Hx63)
(CS, 74, , C, S, k)
(k, 81, , null, null, D;jJ)(4-5Hx63q9L)
(CSk, 155, , CS, k, D;jJ)(4-5Hx63q9LIPB)
(U8'NWCSk, 275, , U8'NW, CSk, v)
(bU8'NWCSk, 546, , b, U8'NWCSk, vw)
(v, 323, , null, null, w)
(w, 338, , null, null, u)
(vw, 661, , v, w, l)
(bU8'NWCSkvw, 1207, , bU8'NWCSk, vw, r)
(ibU8'NWCSkvw, 2338, , i, bU8'NWCSkvw, rh)
(r, 1226, , null, null, h)
(h, 1240, , null, null, n)
(rh, 2466, , r, h, e)
(ibU8'NWCSkvwrh, 4804, , ibU8'NWCSkvw, rh, enluD;jJ)(4-5Hx63q9LIPB.)
(e, 2547, , null, null, nluD;jJ)(4-5Hx63q9LIPB.)
(n, 1321, , null, null, luD;jJ)(4-5Hx63q9LIPB.)
(l, 662, , null, null, uD;jJ)(4-5Hx63q9LIPB.)
(u, 343, , null, null, D;jJ)(4-5Hx63q9LIPB.)
(D, 1, , null, null, ;)
(;, 1, , null, null, 4)
(D;, 2, , D, ;, jJ)
(j, 1, , null, null, J)
(J, 1, , null, null, D)
(jJ, 2, , j, J, 5)
(D;jJ, 4, , D;, jJ, )(4-)
(), 1, , null, null, ()
((, 1, , null, null, )()
()(, 2, , ), (, 4-)
(4, 1, , null, null, -)
(-, 1, , null, null, ))
(4-, 2, , 4, -, D;)
()(4-, 4, , )(, 4-, 5H)
(D;jJ)(4-, 8, , D;jJ, )(4-, 5Hx)
(5, 2, , null, null, H)
(H, 3, , null, null, D;jJ)
(5H, 5, , 5, H, x)
(x, 6, , null, null, O)
(5Hx, 11, , 5H, x, q)
(D;jJ)(4-5Hx, 19, , D;jJ)(4-, 5Hx, 63)
(6, 11, , null, null, 3)
(3, 11, , null, null, OM)
(63, 22, , 6, 3, q9)
(D;jJ)(4-5Hx63, 41, , D;jJ)(4-5Hx, 63, q9L)
(q, 11, , null, null, 9)
(9, 11, , null, null, 6)
(q9, 22, , q, 9, L)
(L, 22, , null, null, P)
(q9L, 44, , q9, L, I)
(D;jJ)(4-5Hx63q9L, 85, , D;jJ)(4-5Hx63, q9L, IPB)
(I, 44, , null, null, PB)
(P, 23, , null, null, B)
```

```
(B,  23,  ,  null,  null,  1)
(PB,  46,  ,  P,  B,  G)
(IPB,  90,  ,  I,  PB,  G1OMF)
(D;jJ)(4-5Hx63q9LIPB,  175,  ,  D;jJ)(4-5Hx63q9L,  IPB,  .)
(.,  181,  ,  null,  null,  G1OMFAT)
(D;jJ)(4-5Hx63q9LIPB.,  356,  ,  D;jJ)(4-5Hx63q9LIPB,  .,  ,)
(uD;jJ)(4-5Hx63q9LIPB.,  699,  ,  u,  D;jJ)(4-5Hx63q9LIPB.,  ,\r)
(luD;jJ)(4-5Hx63q9LIPB.,  1361,  ,  l,  uD;jJ)(4-5Hx63q9LIPB.,  o)
(nluD;jJ)(4-5Hx63q9LIPB.,  2682,  ,  n,  luD;jJ)(4-5Hx63q9LIPB.,  o,\rNLG1OMFATp)
(enluD;jJ)(4-5Hx63q9LIPB.,  5229,  ,  e,  nluD;jJ)(4-5Hx63q9LIPB.,  o,\rNLG1OMFATpagcs)
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.,  10033,  ,  ibU8'NWCSkvwrh,  enluD;jJ)(4-5Hx63q9LIPB.,
o,\rNLG1OMFATpagcsdfymt'space')
(o,  1436,  ,  null,  null,  ,\rNLG1OMFATp)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(,\r,  772,  ,  ,,  \r,  NLG1OMFATp)
(NL,  399,  ,  null,  null,  G1OMFATp)
(G,  46,  ,  null,  null,  1OMF)
(1,  23,  ,  null,  null,  OMF)
(O,  6,  ,  null,  null,  M)
(M,  6,  ,  null,  null,  D;jJ)(4-)
(OM,  12,  ,  O,  M,  F)
(F,  12,  ,  null,  null,  8)
(OMF,  24,  ,  OM,  F,  U)
(1OMF,  47,  ,  1,  OMF,  A)
(G1OMF,  93,  ,  G,  1OMF,  AT)
(A,  47,  ,  null,  null,  T)
(T,  49,  ,  null,  null,  U8')
(AT,  96,  ,  A,  T,  U8'NW)
(G1OMFAT,  189,  ,  G1OMF,  AT,  p)
(p,  216,  ,  null,  null,  y)
(G1OMFATp,  405,  ,  G1OMFAT,  p,  g)
(NLG1OMFATp,  804,  ,  NL,  G1OMFATp,  gc)
(,\rNLG1OMFATp,  1576,  ,  ,\r,  NLG1OMFATp,  a)
(o,\rNLG1OMFATp,  3012,  ,  o,  ,\rNLG1OMFATp,  agcs)
(a,  1627,  ,  null,  null,  gcs)
(g,  406,  ,  null,  null,  c)
(c,  456,  ,  null,  null,  f)
(gc,  862,  ,  g,  c,  s)
(s,  862,  ,  null,  null,  d)
(gcs,  1724,  ,  gc,  s,  dfym)
(agcs,  3351,  ,  a,  gcs,  dfymt)
(o,\rNLG1OMFATpagcs,  6363,  ,  o,\rNLG1OMFATp,  agcs,  dfymt'space')
(d,  876,  ,  null,  null,  fym)
(f,  478,  ,  null,  null,  ym)
(y,  260,  ,  null,  null,  m)
(m,  269,  ,  null,  null,  b)
(ym,  529,  ,  y,  m,  bU8'NWCSk)
(fym,  1007,  ,  f,  ym,  i)
(dfym,  1883,  ,  d,  fym,  t)
(t,  1914,  ,  null,  null,  ibU8'NWCSkvw)
(dfymt,  3797,  ,  dfym,  t,  'space')
('space',  4253,  ,  null,  null,  ibU8'NWCSkvwrh)
(dfymt'space',  8050,  ,  dfymt,  'space',  ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.)
(o,\rNLG1OMFATpagcsdfymt'space',  14413,  ,  o,\rNLG1OMFATpagcs,  dfymt'space',
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space')
(ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.o,\rNLG1OMFATpagcsdfymt'space',  24446,  ,
ibU8'NWCSkvwrhenluD;jJ)(4-5Hx63q9LIPB.,  o,\rNLG1OMFATpagcsdfymt'space',  null)
```

Below is deBugFile:

```
Entering computeCharCounts method!
Leaving computeCharCounts method!
Entering constructHuffmanLList method!
(NL,  399,  ,  null,  null,  null)
Entering findSpot method!
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  null)
(\r,  399,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 399 and newNode's frequency is 399
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  null)
('space',  4253,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 399 and newNode's frequency is 4253
In findSpot: Spot.next's frequency is 399 and newNode's frequency is 4253
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(',  15,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 399 and newNode's frequency is 15
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  ')
(',  15,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
((,  1,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 1
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(),  1,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 1
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(,,  373,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 373
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 373
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 373
In findSpot: Spot.next's frequency is 399 and newNode's frequency is 373
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(-,  1,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 1
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(.,  181,  ,  null,  null,  null)
```

```
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 181
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 181
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 181
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 181
In findSpot: Spot.next's frequency is 373 and newNode's frequency is 181
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(1,  23,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 23
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 23
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 23
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 23
In findSpot: Spot.next's frequency is 181 and newNode's frequency is 23
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  ')
(',  15,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(3,  11,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 11
Leaving findSpot method!
Printing list in constructHuffmanLList method.
```

```
(dummy,  0,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  ')
(',  15,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(4,  1,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 1
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  4)
(4,  1,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  ')
(',  15,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(5,  2,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 2
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 2
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 2
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 2
In findSpot: Spot.next's frequency is 11 and newNode's frequency is 2
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  4)
(4,  1,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  5)
(5,  2,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  ')
```

```
(',  15,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(6,  11,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 2 and newNode's frequency is 11
In findSpot: Spot.next's frequency is 11 and newNode's frequency is 11
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  4)
(4,  1,  ,  null,  null,  -)
(-,  1,  ,  null,  null,  ))
(),  1,  ,  null,  null,  ()
((,  1,  ,  null,  null,  5)
(5,  2,  ,  null,  null,  6)
(6,  11,  ,  null,  null,  3)
(3,  11,  ,  null,  null,  ')
(',  15,  ,  null,  null,  1)
(1,  23,  ,  null,  null,  .)
(.,  181,  ,  null,  null,  ,)
(,,  373,  ,  null,  null,  \r)
(\r,  399,  ,  null,  null,  NL)
(NL,  399,  ,  null,  null,  'space')
('space',  4253,  ,  null,  null,  null)
(8,  13,  ,  null,  null,  null)
Entering findSpot method!
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 1 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 2 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 11 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 11 and newNode's frequency is 13
In findSpot: Spot.next's frequency is 15 and newNode's frequency is 13
Leaving findSpot method!
Printing list in constructHuffmanLList method.
(dummy,  0,  ,  null,  null,  4)
(4,  1,  ,  null,  null,  -)
```