***** PROJECT 5 Hardcopy (PDF File) *****
****** Cover Page ******

Class: CSCI 323 MW
Name: Adewole Adeoshun
Project: Project 5
Project Name: Huffman Coding (Part 1 & Part 2)
Language: Java
Due Date: 10/24/2025, Friday before midnight (11:59 PM)
Submit Date: 10/24/2025

Top-Level Algorithm Steps

Part 1 – Huffman Coding Construction
Step 0: Open args[] files: HuffmanCodingData.txt, outFile.txt, logFile.txt
Step 1: computeCharCounts() → Reads input and counts frequency
Step 2: printCountAry() → Writes ASCII and counts
Step 3: constructHuffmanLList() → Builds sorted list of Huffman nodes
Step 4: printList() → Prints linked list
Step 5: constructHuffmanBinTree() → Builds binary tree
Step 6: inOrderTraversal() → Traverses tree in-order
Step 7: constructCharCode() → Assigns codes
Step 8: printCodeTable() → Writes final code table
Part 2 – Encoding and Decoding
Step 9: Ask to compress input text file
Step 10: Encode text to bitstrings
Step 11: Decode bitstrings back to text
Step 12: Print outFile2 character code table
Step 13: Close all files and print success message

Source Code

```
/*

Name: Adewole Adeoshun

Course: CSCI 323 (MonWed 3:10pm - 4:25pm)
```

Instructor: Tsaiyun Phillips

ID: 24081306

Project 5 - Huffman coding (part-2) // continuation of part-1

How to run:

javac AdeoshunA_Project5_Main.java

java AdeoshunA_Project5_Main HuffmanCodingData.txt outFile.txt outFile2.txt logFile.txt logFile2.txt

*/

```
import java.io.*;

import java.util.*;

// ****************************************

// HTreeNode CLASS

// ****************************************

class HTreeNode {

String chStr; // character(s)

int frequency; // frequency count

String code; // Huffman code

HTreeNode next; // pointer to next node (for LList)

HTreeNode left; // left child

HTreeNode right; // right child

HTreeNode(String c, int f, String cd, HTreeNode n, HTreeNode l, HTreeNode r) {

chStr = c;

frequency = f;
```

```
code = cd;
next = n;
left = l;
right = r;
}
// (T.chStr, T.frequency, T.code, T.left's chStr, T.right's chStr, T.next's chStr)
void printNode(PrintWriter ofile) {
String l = (left != null) ? left.chStr : "null";
String r = (right != null) ? right.chStr : "null";
String n = (next != null) ? next.chStr : "null";
ofile.printf("(%s, %d, %s, %s, %s, %s)%n", chStr, frequency, code, l, r, n);
}
}
// ****************************************
// HuffmanTool (data + algorithms for both parts)
// ****************************************
class HuffmanTool {
HTreeNode Root = null;
HTreeNode listHead = new HTreeNode("dummy", 0, "", null, null, null);
int[] charCountAry = new int[256];
String[] codeTable = new String[256];
HuffmanTool() {
```

```
Arrays.fill(charCountAry, 0);

Arrays.fill(codeTable, "");

}

// ========== Part-1 methods ==========

void computeCharCounts(BufferedReader inFile, PrintWriter logFile) throws
IOException {

logFile.println("**** Entering computeCharCounts ****");

int ch;

while ((ch = inFile.read()) != -1) {

if (0 <= ch && ch < 256) charCountAry[ch]++;

}

logFile.println("**** Leaving computeCharCounts ****");

}

// Print only count>0; skip 10,13,23,32 per professor

void printCountAry(PrintWriter outFile) {

outFile.println("**** Below is the character counts Table ****");

outFile.println("Ascii\tcharacter\tcount");

for (int i = 0; i < 256; i++) {

if (charCountAry[i] > 0 && i != 10 && i != 13 && i != 23 && i != 32) {

outFile.printf("%d\t%c\t%d%n", i, (char)i, charCountAry[i]);

}

}

}
```

```
HTreeNode findSpot(HTreeNode listHead, HTreeNode newNode, PrintWriter logFile) {
logFile.println("**** entering findSpot method ****");
HTreeNode Spot = listHead;
while (Spot.next != null && Spot.next.frequency < newNode.frequency) {
Spot = Spot.next;
}
if (Spot.next != null) {
logFile.printf("in findSpot: Spot.next’s frequency is %d and newNode.frequency is
%d%n",
Spot.next.frequency, newNode.frequency);
}
logFile.println("**** leaving findSpot method ****");
return Spot;
}
void listInsert(HTreeNode Spot, HTreeNode newNode) {
newNode.next = Spot.next;
Spot.next = newNode;
}
void printList(HTreeNode listHead, PrintWriter oFile) {
HTreeNode cur = listHead;
while (cur != null) {
cur.printNode(oFile);
cur = cur.next;
```

```
}

}

HTreeNode constructHuffmanLList(HTreeNode listHead, int[] charCountAry,
PrintWriter logFile) {

logFile.println("**** Entering constructHuffmanLList method ****");

for (int index = 0; index < 256; index++) {

if (charCountAry[index] > 0) {

char chr = (char) index;

int frequency = charCountAry[index];

HTreeNode newNode = new HTreeNode(String.valueOf(chr), frequency, "", null, null,
null);

logFile.println("**** In construct LList, printing newNode ****");

newNode.printNode(logFile);

HTreeNode Spot = findSpot(listHead, newNode, logFile);

listInsert(Spot, newNode);

logFile.println("**** In construct LList, printing list after inserting newNode ****");

printList(listHead, logFile);

}

}

logFile.println("**** Leaving constructHuffmanLList method ****");

return listHead;

}
```

```
HTreeNode constructHuffmanBinTree(HTreeNode listHead, PrintWriter outFile,
PrintWriter logFile) {

logFile.println("**** Entering constructHuffmanBinTree() method ****");

while (listHead.next != null && listHead.next.next != null) {

HTreeNode first = listHead.next;

HTreeNode second = first.next;

HTreeNode newNode = new HTreeNode(

first.chStr + second.chStr,

first.frequency + second.frequency,

"", null, first, second);

logFile.println("**** In constructHuffmanBinTree, printing newNode ****");

newNode.printNode(logFile);

HTreeNode Spot = findSpot(listHead, newNode, logFile);

listInsert(Spot, newNode);

// remove first two nodes

listHead.next = second.next;

logFile.println("**** In constructHuffmanBinTree method, printing the list after
inserting newNode ****");

printList(listHead, logFile);

}

Root = listHead.next;

return Root;

}
```

```
boolean isLeaf(HTreeNode node) {
return node != null && node.left == null && node.right == null;
}
void inOrderTraversal(HTreeNode T, PrintWriter outFile) {
if (T == null) return;
if (isLeaf(T)) {
T.printNode(outFile);
} else {
inOrderTraversal(T.left, outFile);
T.printNode(outFile);
inOrderTraversal(T.right, outFile);
}
}
void constructCharCode(HTreeNode T, String code, String[] codeTable) {
if (T == null) return;
if (isLeaf(T)) {
T.code = code;
int index = (int) T.chStr.charAt(0);
codeTable[index] = code;
} else {
constructCharCode(T.left, code + "0", codeTable);
constructCharCode(T.right, code + "1", codeTable);
```

```
}
}
// Print only characters that actually have a code (non-blank)
void printCodeTable(String[] codeTable, PrintWriter outFile) {
outFile.println("**** Below is the character code Table ****");
outFile.println("Ascii\tcharacter\tcode");
for (int i = 0; i < 256; i++) {
if (!codeTable[i].equals("")) {
outFile.printf("%d\t%c\t%s%n", i, (char)i, codeTable[i]);
}
}
}
// ========== Part-2 methods ==========
// X. userInterface (Root, codeTable, logFile2)
void userInterface(HTreeNode Root, String[] codeTable, PrintWriter logFile2) {
Scanner sc = new Scanner(System.in);
while (true) {
System.out.print("Do you want to compress a text file? (Y for yes, N for no): ");
String ans = sc.nextLine().trim();
if (ans.length() == 0) continue;
char yesNo = Character.toUpperCase(ans.charAt(0));
if (yesNo == 'N') {
```

```
System.out.println("Program exits.");
return;
}
// name without .txt
System.out.print("Enter file name to encode (without .txt), e.g., test1 : ");
String nameOrg = sc.nextLine().trim();
String orgPath = nameOrg + ".txt";
String nameCompress = nameOrg + "_Compressed.txt";
String nameDeCompress = nameOrg + "_deCompressed.txt";
try (BufferedReader orgFile = new BufferedReader(new FileReader(orgPath));
PrintWriter compFile = new PrintWriter(new FileWriter(nameCompress));
PrintWriter deComp = new PrintWriter(new FileWriter(nameDeCompress))) {
Encode(orgFile, compFile, codeTable, logFile2);
compFile.flush();
// reopen compressed as input for Decode
try (BufferedReader compIn = new BufferedReader(new FileReader(nameCompress))) {
Decode(compIn, deComp, Root, logFile2);
}
System.out.println("Created: " + nameCompress + " and " + nameDeCompress);
} catch (IOException ioe) {
System.out.println("I/O Error: " + ioe.getMessage());
}
```

```
}

}

// XI. Encode (FileIn, FileOut, codeTable, logFile2)

void Encode(BufferedReader FileIn, PrintWriter FileOut, String[] codeTable, PrintWriter
logFile2) throws IOException {

logFile2.println("**** Entering Encode method ****");

int ch;

while ((ch = FileIn.read()) != -1) {

int index = (int) ch; // cast to integer

String code = codeTable[index]; // lookup code

// Even if code is empty (shouldn't happen if counted), we still write nothing.

FileOut.print(code); // write code to compressed file

logFile2.printf("** inside Encode() charIn=%d code=%s%n", index, code);

}

logFile2.println("**** Leaving Encode method ****");

}

// XII. Decode (FileIn, FileOut, Root, logFile2)

void Decode(BufferedReader FileIn, PrintWriter FileOut, HTreeNode Root, PrintWriter
logFile2) throws IOException {

logFile2.println("**** Entering Decode method ****");

HTreeNode Spot = Root;

int ch;

while ((ch = FileIn.read()) != -1) {
```

```
char oneBit = (char) ch; // should be '0' or '1'
if (isLeaf(Spot)) {
// write the leaf symbol, reset to root for next symbol
FileOut.write(Spot.chStr);
logFile2.println("Inside Decode method: Spot.chStr=" + Spot.chStr);
Spot = Root;
}
if (oneBit == '0') {
Spot = Spot.left;
} else if (oneBit == '1') {
Spot = Spot.right;
} else {
logFile2.println("Error! The compressed file contains invalid character!");
return;
}
// If we just moved onto a leaf, immediately output and reset
if (isLeaf(Spot)) {
FileOut.write(Spot.chStr);
logFile2.println("Inside Decode method: Spot.chStr=" + Spot.chStr);
Spot = Root;
}
}
```

```
// End of file. If Spot is not root and not a leaf, it ended mid-symbol.
if (Spot != Root && !isLeaf(Spot)) {
logFile2.println("Error: The compress file is corrupted!");
}
logFile2.println("**** Leaving Decode method ****");
}
}
// ****************************************
// MAIN
// ****************************************
public class AdeoshunA_Project5_Main {
public static void main(String[] args) {
try {
if (args.length != 5) {
System.out.println("Usage:");
System.out.println(" java AdeoshunA_Project5_Main inFile outFile outFile2 logFile logFile2");
return;
}
BufferedReader inFile = new BufferedReader(new FileReader(args[0]));
PrintWriter outFile = new PrintWriter(new FileWriter(args[1])); // part-1
PrintWriter outFile2 = new PrintWriter(new FileWriter(args[2])); // part-2
PrintWriter logFile = new PrintWriter(new FileWriter(args[3])); // part-1
```

```
PrintWriter logFile2 = new PrintWriter(new FileWriter(args[4])); // part-2
HuffmanTool tool = new HuffmanTool();
// ======== Part 1 sequence ========
tool.computeCharCounts(inFile, logFile); inFile.close();
tool.printCountAry(outFile);
outFile.println("*** In main, calling constructHuffmanLList ***");
tool.listHead = tool.constructHuffmanLList(tool.listHead, tool.charCountAry, logFile);
outFile.println("*** In main (): printing list after list is constructed ***");
tool.printList(tool.listHead, outFile);
outFile.println("*** In main before calling constructHuffmanBinTree ***");
tool.Root = tool.constructHuffmanBinTree(tool.listHead, outFile, logFile);
outFile.println("*** In main (): printing in-Order traversal of the tree ***");
tool.inOrderTraversal(tool.Root, outFile);
tool.constructCharCode(tool.Root, "", tool.codeTable);
outFile.println("*** In main (): printing character code Table ****");
tool.printCodeTable(tool.codeTable, outFile);
// ======== Part 2 sequence ========
outFile2.println("*** In main (part-2): printing character code Table ****");
tool.printCodeTable(tool.codeTable, outFile2);
tool.userInterface(tool.Root, tool.codeTable, logFile2);
outFile.close();
outFile2.close();
```

```
logFile.close();

logFile2.close();

System.out.println("Project Completed.");

} catch (IOException ioe) {

System.out.println("I/O Error: " + ioe.getMessage());

}

}

}
```

***** Below is outFile *****

**** Below is the character counts Table ****

Ascii	character	count
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
*** In main, calling constructHuffmanLList ***
*** In main (): printing list after list is constructed ***
(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,
)
(
, 398, , 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,)
(, 4253, , null, null, null)
*** In main before calling constructHuffmanBinTree ***
*** In main (): printing in-Order traversal of the tree ***
(y, 260, , null, null, m)
(ym, 529, , y, m, bU8'NWCSk)
(m, 269, , null, null, b)
(ymbU8'NWCSk, 1075, , ym, bU8'NWCSk, i)
(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)
(ymbU8'NWCSki, 2206, , ymbU8'NWCSk, i, rh)
(i, 1131, , null, null, r)
(ymbU8'NWCSkirh, 4672, , ymbU8'NWCSki, rh, envwl)
(r, 1226, , null, null, h)
(rh, 2466, , r, h, e)
(h, 1240, , null, null, n)
(ymbU8'NWCSkirhenvwl, 9863, , ymbU8'NWCSkirh, envwl, ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgsdcft)
(e, 2547, , null, null, nvwl)
(envwl, 5191, , e, nvwl, ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgs)
(n, 1321, , null, null, vwl)
(nvwl, 2644, , n, vwl, ouD;jJ)(4-5Hx63q9LIPB.,
)
(v, 323, , null, null, w)
(vw, 661, , v, w, l)
(w, 338, , null, null, u)
(vwl, 1323, , vw, l, o)
(l, 662, , null, null, uD;jJ)(4-5Hx63q9LIPB.)
(ymbU8'NWCSkirhenvwlouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgsdcft , 24046, , ymbU8'NWCSkirhenvwl, ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgsdcft , null)
(o, 1436, , null, null, uD;jJ)(4-5Hx63q9LIPB.,

)
(ouD;jJ)(4-5Hx63q9LIPB.,
, 2906, , o, uD;jJ)(4-5Hx63q9LIPB.,
, aG1OMFATpgs)
(u, 343, , null, null, D;jJ)(4-5Hx63q9LIPB.)
(uD;jJ)(4-5Hx63q9LIPB., 699, , u, D;jJ)(4-5Hx63q9LIPB., ,
)
(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)
(uD;jJ)(4-5Hx63q9LIPB.,
, 1470, , uD;jJ)(4-5Hx63q9LIPB., ,
, a)
(,, 373, , null, null,
)
(,
, 771, , ,,
, G1OMFATpg)
(
, 398, , null, null, G1OMFATp)
(ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgs, 6206, , ouD;jJ)(4-5Hx63q9LIPB.,
, aG1OMFATpgs, dcft)
(a, 1627, , null, null, G1OMFATpgs)
(aG1OMFATpgs, 3300, , a, G1OMFATpgs, dcft)
(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)
(G1OMFATpg, 811, , G1OMFATp, g, s)
(g, 406, , null, null, c)
(G1OMFATpgs, 1673, , G1OMFATpg, s, dcf)
(s, 862, , null, null, d)
(ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgsdcft , 14183, , ouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgs, dcft , ymbU8'NWCSkirhenvwlouD;jJ)(4-5Hx63q9LIPB.,
aG1OMFATpgsdcft)
(d, 876, , null, null, cf)
(dcf, 1810, , d, cf, t)
(c, 456, , null, null, f)
(cf, 934, , c, f, ymbU8'NWCSk)
(f, 478, , null, null, ym)
(dcft, 3724, , dcf, t,)
(t, 1914, , null, null, ymbU8'NWCSki)
(dcft , 7977, , dcft, , ymbU8'NWCSkirhenvwl)
(, 4253, , null, null, ymbU8'NWCSkirh)
*** In main (): printing character code Table ****
**** Below is the character code Table ****

Ascii	character	code
10		
	100111	
32		111
39	'	0000110011
40	(	10010100000101
41	)	10010100000100
44	,	100110
45	-	10010100000111
46	.	1001011
49	1	1011000010
51	3	10010100011
52	4	10010100000110
53	5	1001010000100
54	6	10010100010
56	8	0000110010

57	9	10010100101
59	;	10010100000001
65	A	101100010
66	B	1001010111
67	C	000011100
68	D	10010100000000
70	F	10110000111
71	G	101100000
72	H	1001010000101
73	I	100101010
74	J	10010100000011
76	L	1001010011
77	M	101100001101
78	N	000011010
79	O	101100001100
80	P	1001010110
83	S	000011101
84	T	101100011
85	U	000011000
87	W	000011011
97	a	1010
98	b	000010
99	c	110010
100	d	11000
101	e	010
102	f	110011
103	g	101101
104	h	0011
105	i	0001
106	j	10010100000010
107	k	00001111
108	l	01111
109	m	000001
110	n	0110
111	o	1000
112	p	1011001
113	q	10010100100

114	r	0010
115	s	10111
116	t	1101
117	u	100100
118	v	011100
119	w	011101
120	x	100101000011
121	y	000000

***** Below is test1 *****

The boy visits Santiago's shack each night,
hauling his fishing gear, preparing food,
talking about American baseball and his favorite player,
Joe DiMaggio. Santiago tells Manolin that on the next day,
he will venture far out into the Gulf Stream,
north of Cuba in the Straits of Florida to fish,
confident that his unlucky streak is near its end.

***** Below is test1_Compressed *****

1011000110011010111000010100000000001110111000001101110001110110111110000
1110110100110110100011010101101100000001100111011111110111001110101100100
0001111111010101011001000111110110000110110100111011001101111001110011110
1010010001111000101101011011110011000110111111110011000110111001100010110
1011011111011010101010001010011011110110010010101011001101000100001011 01
0110111111001110001000110001001101111001111101101001111000011110001011010
1101111101000001010001001001101111101100010000001010001000011100101010011
0111000010101010111010000010101001111011111111010011011000111001100011011
1111110011101001110010000010000111010101111011001011111010000000010001010
0110111100111100101000000111000010111100101000000000000110110000110110101 0
1101101101000110001001011111000011101101001101101000110101011011000111110
1010011110111110111111101100001101101001101000011110001011011111010011101 0
1101111100001101111101001101011101100101001010000111101111111000101000000 0

100110111100111001101011101110100010111101111111011100010011011011000010000
100101111100111010001011110001001001101111000101101101100011111010011010 1
111011000001001000111111001111100001110111010010010101000000110011011110 0
111011010000010110100111111000110011111000011100100100000010101011100010 1
101111101001101011100001110111010010101000011101101111111000110011111101 10
000111011111000001000011100010101111101100011111001100011011100111001101 1
110011111001010000110110011000111000010011011011111101001110101101111001 1
000110111111100100011001111100100110010000011110000000111011111010010010 1
010000011111110001101111110110010101000101110001110110111111010011011000 1
001011111

***** Below is test1_deCompressed *****

The boy visits Santiago's shack each night,
hauling his fishing gear, preparing food,
talking about American baseball and his favorite player,
Joe DiMaggio. Santiago tells Manolin that on the next day,
he will venture far out into the Gulf Stream,
north of Cuba in the Straits of Florida to fish,
confident that his unlucky streak is near its end.

***** Below is outFile2 for test1 *****

*** In main (part-2): printing character code Table ****
**** Below is the character code Table ****

Ascii	character	code
10		100111
32		111
39	'	0000110011
40	(	10010100000101
41	)	10010100000100
44	,	100110
45	-	10010100000111
46	.	1001011
49	1	1011000010

51	3	10010100011
52	4	10010100000110
53	5	1001010000100
54	6	10010100010
56	8	0000110010
57	9	10010100101
59	;	10010100000001
65	A	101100010
66	B	1001010111
67	C	000011100
68	D	10010100000000
70	F	10110000111
71	G	101100000
72	H	1001010000101
73	I	100101010
74	J	10010100000011
76	L	1001010011
77	M	101100001101
78	N	000011010
79	O	101100001100
80	P	1001010110
83	S	000011101
84	T	101100011
85	U	000011000
87	W	000011011
97	a	1010
98	b	000010
99	c	110010
100	d	11000
101	e	010
102	f	110011
103	g	101101
104	h	0011
105	i	0001
106	j	10010100000010
107	k	00001111
108	l	01111

109	m	000001
110	n	0110
111	o	1000
112	p	1011001
113	q	10010100100
114	r	0010
115	s	10111
116	t	1101
117	u	100100
118	v	011100
119	w	011101
120	x	100101000011
121	y	000000

***** Below is test2 *****

The Gettysburg Address is a speech by U.S. President Abraham Lincoln,
one of the best known in American history.
It was delivered by Lincoln during the American Civil War,
on the afternoon of Thursday, November 19, 1863, at the dedication of the Soldiers'
National Cemetery in Gettysburg, Pennsylvania,
four and a half months after the Union armies defeated those of the Confederacy
at the Battle of Gettysburg.

***** Below is test2_Compressed *****

1011000110011010111101100000010110111010000001011100001010010000101011011
11101100010110001100000100101011110111111000110111111101011110111101100101
00101100100011111000010000000011100001100010010110000111011001011111100101
0110001001010111000111000010011011011111011000100000100010101000111010000
0011111001010011000101101100101000011110110100110111100111100001100101111
000110011111101001101011100001001010111110111100001111011010000111010110
1110001011011110110001000000101000100001110010101001101110011000110111110
11000001000000001001011111100111100101010110111101110110101011111111000010

01111000101110001000100101100011100001000000011110010100110001011011001010
0000111101101111100010010000100001011010110111111010011010111101100010000
00101000100001110010101001101110000111000001011100000101111111000011011100
1000101001101111001111000011011111010011010111101011001111010100010011010
00100000110111100011001111110110001100111001000010101111100010100000001001
10111000011010100001110001000000100001001000101111011000010100101001011010
01101111011000010000011001010010100010100101000111001101111010110111111101
00110101111100001011000000111001010101101000110000110111100011001111111101
00110101110000111011000011111100000010100010101110000110011111100111000011
10101010110100011000011010100111111100001110001000000101011010100010000000
00111000101101111011000000101101110100000010111000010100100001010110110011
10111100101011001001100110101110000000111101110010100110000110101001101111
1001111110011100010010000101111010011011000111101011100111010011111100111111
00000110000110110100111011111110101100111101010001011111010011010111000011
10000110000110000110111101000100000001000101010111111110000101100110101010
11010101100011111010011100010111010111100011001111111010011010111000011100
01000011011001101011000010001010101100100000001111001111010110111111010011
1010111100101011110101101110101111010111100011001111110110000000101101110
1000000010111000010100100001010110110010111111

***** Below is test2_deCompressed *****

The Gettysburg Address is a speech by U.S. President Abraham Lincoln,
one of the best known in American history.
It was delivered by Lincoln during the American Civil War,
on the afternoon of Thursday, November 19, 1863, at the dedication of the Soldiers'
National Cemetery in Gettysburg, Pennsylvania,
four and a half months after the Union armies defeated those of the Confederacy
at the Battle of Gettysburg.

***** Below is outFile2 for test2 *****

*** In main (part-2): printing character code Table ****
**** Below is the character code Table ****
Ascii character code
10

	100111	
32		111
39	'	0000110011
40	(	10010100000101
41	)	10010100000100
44	,	100110
45	-	10010100000111
46	.	1001011
49	1	1011000010
51	3	10010100011
52	4	10010100000110
53	5	1001010000100
54	6	10010100010
56	8	0000110010
57	9	10010100101
59	;	10010100000001
65	A	101100010
66	B	1001010111
67	C	000011100
68	D	10010100000000
70	F	10110000111
71	G	101100000
72	H	1001010000101
73	I	100101010
74	J	10010100000011
76	L	1001010011
77	M	101100001101
78	N	000011010
79	O	101100001100
80	P	1001010110
83	S	000011101
84	T	101100011
85	U	000011000
87	W	000011011
97	a	1010
98	b	000010
99	c	110010

100	d	11000
101	e	010
102	f	110011
103	g	101101
104	h	0011
105	i	0001
106	j	10010100000010
107	k	00001111
108	l	01111
109	m	000001
110	n	0110
111	o	1000
112	p	1011001
113	q	10010100100
114	r	0010
115	s	10111
116	t	1101
117	u	100100
118	v	011100
119	w	011101
120	x	100101000011
121	y	000000

***** Below is test3 *****

Santiago straps the marlin to the side of his skiff and heads home,
thinking about the high price the fish will bring him at the market
and how many people he will feed.
On his way in to shore, sharks are attracted to the marlin's blood.
Santiago kills a great mako shark with his harpoon,
but he loses the weapon.
He makes a new harpoon by strapping his knife
to the end of an oar to help ward off the next line of sharks;
five sharks are slain and many others are driven away.
But the sharks keep coming, and by nightfall

the sharks have almost devoured the marlin's entire carcass,
leaving a skeleton consisting mostly of its backbone, its tail,
and its head. Santiago knows that he is destroyed and
tells the sharks of how they have killed his dreams.

***** Below is test3_Compressed *****

000011101101001101101000110101011011000111101111101001010101011001101111
111010011010111000001101000100111100010110111110110001111101001101011101
110001110000101111000110011111001100011011111110111000011110001110011111001
111110100110110001110011010101011000101111110011100000000101010011011110 0
111110100110001011000001111000101101011011111010000010100010010011011111 1
010011010111001100011011010011111101100100100001110010010111110100110101 1
111001100011011100111110111010001011110111111100001000100001011010110111 1
001100010000011111010110111111010011010111000001101000100000111101011011 1
110011110100110110001110011100000111011110000011010011000000011110110010 10
100010110010111101011100110101110111010001011110111111111001101001011000 1
001011111100111101100001100011011100110001101111110111011010000000111000 1
011011111011000111101110011100000100101001101111011100111010001000001111 1
011111110100010010111101011011101001010101100101101010110001111101100011 1
110100110101110000011010001001111000101100000110011101111110000100111110 0
010001100010010111111001110000111011010011011010001101010110110000111000 01
111000101111011111011111110101111011010010010101011011110000011010000011 11
100011110111001110100010000011111111011101000111010011111001100011011111100
111010001010110011000100001101001101111001110000101001001101111001101011 1
011111000101110101011111111010011010111011101010101010110011000011010010 1
111110011110010100001010101110000011010000011110101011111110101110110010 0
111011110011101000101011001100010000110111000010000000111101111101001010 1
010110011011001000101101011011110011000110111111000011110110000111001101 0
111001111101100011111010011010111010011011000111100011001111110100110111 1
000101000101111101100011100110100111110110011110111011010001011000111100 0
110011110011111110100110101110110010100101000011110111101111000101100101 1
110001100111111011100111010001000001111101111001010000000111110011111001 10
001011100010111101110011101000100000111110111111010001001011110111011111 0
100001011011110100110110001110000011010011000000011110001101001101000101 0
111111101000100101111100000100001011100010011011110100111011010000000100 1

01111110011110010101111001001101111110100110101111011100111010001000001111
10111111000011110100101011001111110010100000000100010110101101100110111110
10011011000111000010000000111011000011011010011110111001110100111101111111
11001111101001101011110111001110100010000011111011111100111010011100010111
10100111100000110001011111011111100001001110010001001000010010110001111110
10011010111000001101000100111100010110000011001110111111010011011010001000
10010111110010101000101100101010101111011110011011110011101111010101001111
00000101101011011111010111011100001111010011110101101100001101111100101000
00011010111000110111110100010110101101111000001100010111110101111000000011
11000110011111000111011011111100001010101100100000111100001010000110010100
01101110001110110111111110110100001011111001101111001111010011011000111000
11101101111110011010101011000100101111100001110110100110110100011010101100
11000111000011110110100001110110111111110100111010110111100110101110001101
11111110000101011111010010100000000001011000111101001101100011110011111101
01001111011111011111111010011010111101110011101000100000111110111111100011
00111110011100001110111111010011010000000111001110100111000101110000111100
00101111011110101100011100110001101111111000001001010100000011011110010111
1111

***** Below is test3_deCompressed *****

Santiago straps the marlin to the side of his skiff and heads home,
thinking about the high price the fish will bring him at the market
and how many people he will feed.
On his way in to shore, sharks are attracted to the marlin's blood.
Santiago kills a great mako shark with his harpoon,
but he loses the weapon.
He makes a new harpoon by strapping his knife
to the end of an oar to help ward off the next line of sharks;
five sharks are slain and many others are driven away.
But the sharks keep coming, and by nightfall
the sharks have almost devoured the marlin's entire carcass,
leaving a skeleton consisting mostly of its backbone, its tail,
and its head. Santiago knows that he is destroyed and
tells the sharks of how they have killed his dreams.

***** Below is outFile2 for test3 *****

*** In main (part-2): printing character code Table ****
**** Below is the character code Table ****

Ascii	character	code
10		
	100111	
32		111
39	'	0000110011
40	(	10010100000101
41	)	10010100000100
44	,	100110
45	-	10010100000111
46	.	1001011
49	1	1011000010
51	3	10010100011
52	4	10010100000110
53	5	1001010000100
54	6	10010100010
56	8	0000110010
57	9	10010100101
59	;	10010100000001
65	A	101100010
66	B	1001010111
67	C	000011100
68	D	10010100000000
70	F	10110000111
71	G	101100000
72	H	1001010000101
73	I	100101010
74	J	10010100000011
76	L	1001010011
77	M	101100001101
78	N	000011010
79	O	101100001100
80	P	1001010110
83	S	000011101
84	T	101100011
85	U	000011000
87	W	000011011
97	a	1010
98	b	000010
99	c	110010
100	d	11000

101	e	010
102	f	110011
103	g	101101
104	h	0011
105	i	0001
106	j	10010100000010
107	k	00001111
108	l	01111
109	m	000001
110	n	0110
111	o	1000
112	p	1011001
113	q	10010100100
114	r	0010
115	s	10111
116	t	1101
117	u	100100
118	v	011100
119	w	011101
120	x	100101000011
121	y	000000
