File Compressor Test Plan

File Compressor Test Plan details the test cases that we ran to check the functionality of our program.
Note: All code blocks start with non-whitespace. The PDF converter added a space in the front sometimes that should not be there.

Build codebook

Test 1
inputtest.txt:

```
 a
a
```

Command:

```
./fileCompressor -b inputtest.txt
```

HuffmanCodebook:

```
 !
0   a
1   !n
```

Success: Only 2 unique tokens, so 0 for one and 1 for the other.
Test 2
inputtest.txt:

```
 !
!
```

Command:

```
./fileCompressor -b inputtest.txt
```

HuffmanCodebook:

```
 !!
0   !!n
1   !
```

Success: Correctly detected "!", our starting escape sequence, and changed escape sequence to "!!" so newline becomes !!n.

Test 3

inputtest.txt:

```
 a   q w     d s     a sd    f d s    s w     f
a q w       d   s   sd w
    !!! !!! w f
```

Command:

```
./fileCompressor -b inputtest.txt
```

HuffmanCodebook:

```
 !!!!
000 w
0010    f
0011    a
01000   !!!!n
01001   sd
0101    d
0110    s
01110   q
01111   !!!
1
```

Success: Changed escape sequence to "!!!!" and correctly assigned shortest codes to most frequenct tokens ' ' and 'w'.

Compress

Test 1

inputtest.txt:

```
 a
a
```

HuffmanCodebook:

```
!
0   a
1   !n
```

Command:

```
./fileCompressor -c inputtest.txt HuffmanCodebook
```

inputtest.txt.hcz:

```
0101
```

Success: Binary code is a correct translation of original file using codebook.

Test 2

inputtest.txt:

```
!
!
```

HuffmanCodebook:

```
!!
0   !!n
1   !
```

Command:

```
./fileCompressor -c inputtest.txt HuffmanCodebook
```

inputtest.txt.hcz:

```
1010
```

Success: Binary code is a correct translation of original file using codebook.

Test 3

inputtest.txt:

```
 a   q w     d s     a sd    f d s    s w     f
a q w       d   s   sd w
    !!! !!! w f
```

HuffmanCodebook:

```
 !!!!
000 w
0010    f
0011    a
01000   !!!!n
01001   sd
0101    d
0110    s
01110   q
01111   !!!
1
```

Command:

```
./fileCompressor -c inputtest.txt HuffmanCodebook
```

inputtest.txt.hcz:

```
 00111111111011101000111110101101101111100111010011111001010101101101111011010001111100101
 11111111111111111111111111111111111111100111011101000111111111110101111111101101111111010
 11111111111111111111111111111111111111111111111111111111111111111111111111011111111101111111111
```

Success: Binary code is a correct translation of original file using codebook.

Decompress

Test 1 inputtest.txt.hcz:

```
0101
```

HuffmanCodebook:

```
!
0   a
1   !n
```

Command:

```
./fileCompressor -d inputtest.txt.hcz HuffmanCodebook
```

inputtest.txt:

```
a
a
```

Success: Decompressed file matches original file.

Test 2

inputtest.txt.hcz:

```
1010
```

HuffmanCodebook:

```
!!
0   !!n
1   !
```

Command:

```
./fileCompressor -d inputtest.txt.hcz HuffmanCodebook
```

inputtest.txt:

```
!
!
```

Success: Decompressed file matches original file.

Test 3

inputtest.txt.hcz:

```
00111111111011101000111110101101101111100111010011111001010101101101111011010001111100101
11111111111111111111111111111111111111110011101110100011111111111010111111110110111111101
11111111111111111111111111111111111111111111111111111111111111111111111011111111101111111111
```

HuffmanCodebook:

```
!!!!
000 w
0010    f
0011    a
01000   !!!!n
01001   sd
0101    d
0110    s
01110   q
01111   !!!
1
```

Command:

```
./fileCompressor -d inputtest.txt.hcz HuffmanCodebook
```

inputtest.txt:

```
a   q w     d s     a sd    f d s    s w     f
a q w       d   s   sd w
    !!! !!! w f
```

Success: Decompressed file matches original file.

Recursive build codebook

Test 1

Directory Structure:

```
recursionTest/
|-- anotherDir
|   |-- dir3
|   |   `-- lel
|   `-- file3
|-- testfile1
`-- testfile2
```

lel:

```
hi
```

file3:

```
another file in another dir
```

testfile1:

```
this is a file
```

testfile2:

```
this is also a file
```

Command:

```
./fileCompressor -R -b recursionTest/
```

HuffmanCodebook:

```
 !
0
1000    this
1001    a
101 !n
11000   dir
11001   also
1101    another
1110    file
11110   is
111110  in
111111  hi
```

Success: All tokens were detected and binary codes generated that reflect their frequencies.

Recursive compress

Test 1

Same directory structure and file content as above.

HuffmanCodebook:

```
 !
0
1000    this
1001    a
101 !n
11000   dir
11001   also
1101    another
1110    file
11110   is
111110  in
111111  hi
```

Command:

```
./fileCompressor -R -c recursionTest/ HuffmanCodebook
```

lel.hcz:

```
111111101
```

file3.hcz:

```
1101011100111110011010110000101
```

testfile1.hcz:

```
10000111100100101110101
```

testfile2.hcz:

```
100001111001100101001011101O1
```

Success: All files found were correctly compressed into binary code.

Recursive decompress

Test 1

Same directory structure and file content as above.
HuffmanCodebook:

```
 !
0
1000    this
1001    a
101 !n
11000   dir
11001   also
1101    another
1110    file
11110   is
111110  in
111111  hi
```

Command:

```
./fileCompressor -R -d recursionTest/ HuffmanCodebook
```

lel:

```
hi
```

file3:

```
another file in another dir
```

testfile1:

```
this is a file
```

testfile2:

```
this is also a file
```

Success: All decompressed files match original files.