**Test 1:**

|  |  |
| --- | --- |
| A | 00 |
| U | 01 |
| S | 101 |
| T | 10001 |
| I | 11010 |
| N | 1001 |
| C | 11101 |
| H | 11001 |
| R | 11000 |
| G | 10000 |

Message:

0001101100011101010011011110111001011100010000

Expected Output:

Success: AUSTINSCHURG

Number of bits = 46

Number of characters = 12

Compression ratio = .479

Why:

This test case was included as another basic, but larger test case to ensure the program worked on a simple valid text file that wasn’t used during the writing and development of the code. Also it prints my name which is cool.

**Test 2:**

|  |  |
| --- | --- |
| C | 00000000000000000010 |
| S | 00011100010001110010100001 |
| I | 011111110000100110011100 |

Message:

000000000000000000100001110001000111001010000100000000000000000010011111110000100110011100

Expected Output:

Success: CSCI

Number of bits = 90

Number of characters = 4

Compression ratio = 2.81

Why:

This test case was used to see if my program could handle long keys for placement in the binary tree. It worked just fine. I also made a variation of this case where I tried adding a 1 to the end of the I key to make sure an error would appear. An error message did appear saying there was something wrong with the message.

**Test 3:**

|  |  |
| --- | --- |
| A | 1001 |
| L | 1021 |

Message:

100110211021

Expected Output:

Error: not a valid code

Why:

Used this test case to see what would happen if user tried a non-binary coding scheme.