sanity_tests.py::test_avg_length_globals_scramble
pass
1 / 1
sanity_tests.py::test_build_frequency_dict_globals_scramble
pass
1 / 1
sanity_tests.py::test_build_huffman_tree_globals_scramble
pass
1 / 1
sanity_tests.py::test_get_codes_globals_scramble
pass
1 / 1
sanity_tests.py::test_improve_tree_globals_scramble
pass
1 / 1
sanity_tests.py::test_number_nodes_globals_scramble
pass
1 / 1
sanity_tests.py::test_tree_to_bytes_globals_scramble
pass
1 / 1
sanity_tests.py::test_avg_length_globals
pass
1 / 1
sanity_tests.py::test_build_frequency_dict_globals
pass
1 / 1
sanity_tests.py::test_build_huffman_tree_globals
pass
1 / 1
sanity_tests.py::test_compress_bytes_globals
pass
1 / 1
sanity_tests.py::test_decompress_bytes_globals
pass
1 / 1
sanity_tests.py::test_generate_tree_general_globals
pass
1 / 1
sanity_tests.py::test_generate_tree_postorder_globals
pass
1 / 1
sanity_tests.py::test_get_codes_globals

pass
1 / 1
sanity_tests.py::test_improve_tree_globals
pass
1 / 1
sanity_tests.py::test_number_nodes_globals
pass
1 / 1
sanity_tests.py::test_tree_to_bytes_globals
pass
1 / 1
sanity_tests.py::test_avg_length_signature
pass
1 / 1
sanity_tests.py::test_build_frequency_dict_signature
pass
1 / 1
sanity_tests.py::test_build_huffman_tree_signature
pass
1 / 1
sanity_tests.py::test_compress_bytes_signature
pass
1 / 1
sanity_tests.py::test_decompress_bytes_signature
pass
1 / 1
sanity_tests.py::test_generate_tree_general_signature
pass
1 / 1
sanity_tests.py::test_generate_tree_postorder_signature
pass
1 / 1
sanity_tests.py::test_get_codes_signature
pass
1 / 1
sanity_tests.py::test_improve_tree_signature
pass
1 / 1
sanity_tests.py::test_number_nodes_signature
pass
1 / 1
sanity_tests.py::test_tree_to_bytes_signature
pass
1 / 1

test_huffman_properties_extra_generate_tree.py::test_generate_tree_general (Test that generate_tree_general keeps the same tree shape as the original tree.)
pass
1 / 1
test_huffman_properties_extra_generate_tree.py::test_generate_tree_postorder (Test that generate_tree_postorder keeps the same tree shape as the original tree.)
fail
0 / 1
test_private_helpers.py::test_helpers_private (Test that all helper functions are private.)
pass
1 / 1
test_number_of_examples.py::test_number_nodes_2 (Test whether the Huffman tree created from the frequency dictionary <d> is numbered correctly, as expected in postorder traversal.)
pass
1 / 1
test_huffman_basic.py::test_01_build_frequency_dict (Test that the build_frequency_dict produces the correct dictionary.)
pass
1 / 1
test_huffman_basic.py::test_02_get_codes (Test whether get_codes produces the correct dictionary that maps symbols from the Huffman tree to codes.)
pass
1 / 1
test_huffman_basic.py::test_03_number_nodes (Test whether number_nodes correctly numbers the nodes in a Huffman tree.)
pass
1 / 1
test_huffman_basic.py::test_04_avg_length (Test if the average number of bits required per symbol returned by avg_length is correct, within a reasonable floating point precision.)
pass
1 / 1
test_huffman_basic.py::test_05_compress_bytes (Test if compress_bytes produces the correct compressed text.)
pass
1 / 1
test_huffman_basic.py::test_06_tree_to_bytes (Test if the bytes representation of the Huffman tree, as given by tree_to_bytes, is correct.)
pass
1 / 1
test_huffman_basic.py::test_07_generate_tree_general (Test if the generate_tree_general produces the correct Huffman tree.)
pass
1 / 1

test_huffman_basic.py::test_08_generate_tree_postorder (Test if the generate_tree_postorder produces the correct Huffman tree.)
fail
0 / 1
test_huffman_basic.py::test_09_decompress_bytes (Test if the decompress_bytes produces the correct text.)
pass
1 / 1
test_huffman_basic.py::test_10_build_huffman_tree (Test if build_huffman_tree produces a Huffman tree equivalent to the instructor's.)
pass
1 / 1
test_huffman_basic.py::test_11_improve_tree (Test if improve_tree produces a tree equivalent to the instructor's.)
fail
0 / 1
test_huffman_properties_basic_compress_decompress.py::test_compress_bytes (Test that compress_bytes returns a bytes object that is no longer than the input bytes. Also, the size of the compressed object should be invariant under permuting the input. Note: this also indirectly tests build_frequency_dict, build_huffman_tree, and get_codes.)
pass
1 / 1
test_huffman_properties_basic_compress_decompress.py::test_round_trip_compress_bytes (Test that applying compress_bytes and then decompress_bytes will produce the original text.)
pass
1 / 1
test_huffman_properties_extra.py::test_build_frequency_dict (Test that build_frequency_dict has the same keys as the bytes in the text.)
pass
1 / 1
test_huffman_properties_extra.py::test_build_huffman_tree (Test that build_huffman_tree produces a HuffmanTree instance and that the tree has one fewer internal nodes than dict keys.)
pass
1 / 1
test_huffman_properties_extra.py::test_get_codes_same (Test that get_codes has the same length as the freq_dict that was used to generate the Huffman tree.)
pass
1 / 1
test_huffman_properties_extra.py::test_get_codes_length (Test that get_codes values are all strings of length between 1 and 255.)
pass
1 / 1

test_huffman_properties_extra.py::test_number_nodes (Test that number_nodes does not change shape of tree.)
pass
1 / 1
test_huffman_properties_extra.py::test_avg_length (Test that avg_length does not modify its arguments.)
pass
1 / 1
test_huffman_properties_extra.py::test_tree_to_bytes (Test that tree_to_bytes does not modify its argument.)
pass
1 / 1
test_huffman_properties_extra_compress_decompress.py::test_compress_bytes (Test that compress_bytes does not modify its arguments.)
pass
1 / 1
test_huffman_properties_extra_compress_decompress.py::test_decompress_bytes (Test that decompress_bytes is of type bytes and is the same size as the original text.)
pass
1 / 1
test_huffman_properties_basic.py::test_build_frequency_dict (Test that build_frequency_dict returns dictionary whose values sum up to the number of bytes consumed.)
pass
1 / 1
test_huffman_properties_basic.py::test_build_huffman_tree (Test that build_huffman_tree returns a non-leaf HuffmanTree.)
pass
1 / 1
test_huffman_properties_basic.py::test_get_codes (Test that the sum of len(code) * freq_dict[code] is optimal, so it must be invariant under permutation of the dictionary. Note: This also tests build_huffman_tree indirectly.)
pass
1 / 1
test_huffman_properties_basic.py::test_number_nodes (If the root is an interior node, it must be numbered two less than the number of symbols, since a complete tree has one fewer interior nodes than it has leaves, and we are numbering from 0. Note: this also tests build_huffman_tree indirectly.)
pass
1 / 1
test_huffman_properties_basic.py::test_avg_length (Test that avg_length returns a float in the interval [0, 8], if the max number of symbols is 256.)
pass
1 / 1

test_huffman_properties_basic.py::test_compress_bytes (Test that compress_bytes returns a bytes object that is no longer than the input bytes. Also, the size of the compressed object should be invariant under permuting the input. Note: this also indirectly tests build_frequency_dict, build_huffman_tree, and get_codes.)
pass
1 / 1
test_huffman_properties_basic.py::test_tree_to_bytes (Test that tree_to_bytes generates a bytes representation of a postorder traversal of a tree's internal nodes. Since each internal node requires 4 bytes to represent, and there are 1 fewer internal nodes than distinct symbols, the length of the bytes produced should be 4 times the length of the frequency dictionary, minus 4. Note: also indirectly tests build_frequency_dict, build_huffman_tree, and number_nodes.)
pass
1 / 1
test_huffman_properties_basic.py::test_round_trip_compress_bytes (Test that applying compress_bytes and then decompress_bytes will produce the original text.)
pass
1 / 1
test_huffman_extensive_generate_tree_postorder.py::test_generate_tree_postorder (Test using hypothesis if the generate_tree_postorder produces the correct Huffman tree.)
fail
0 / 1
test_huffman_extensive_compress.py::test_compress_bytes (Test using hypothesis if compress_bytes produces the correct compressed text.)
pass
1 / 1
test_huffman_extensive_decompress.py::test_decompress_bytes (Test using hypothesis if the decompress_bytes produces the correct text.)
pass
1 / 1
test_huffman_extensive.py::test_build_frequency_dict (Test using hypothesis that the build_frequency_dict produces the correct dictionary.)
pass
1 / 1
test_huffman_extensive.py::test_get_codes (Test using hypothesis if get_codes produces the correct dictionary that maps symbols from the Huffman tree to codes.)
pass
1 / 1
test_huffman_extensive.py::test_number_nodes (Test using hypothesis if number_nodes correctly numbers the nodes in a Huffman tree.)
pass
1 / 1
test_huffman_extensive.py::test_avg_length (Test using hypothesis if the average number of bits required per symbol, as returned by avg_length, is correct within a reasonable floating point precision.)

pass
1 / 1
test_huffman_extensive.py::test_tree_to_bytes (Test using hypothesis if the bytes representation of the Huffman tree, as given by tree_to_bytes, is correct.)
pass
1 / 1
test_huffman_extensive.py::test_build_huffman_tree (Test using hypothesis if build_huffman_tree produces a Huffman tree equivalent to the instructor's.)
pass
1 / 1
test_huffman_extensive.py::test_build_huffman_tree_single (Test that building a tree with a single symbol returns a valid tree.)
pass
1 / 1
test_huffman_extensive_generate_tree_general.py::test_generate_tree_general (Test using hypothesis if the generate_tree_general produces the correct Huffman tree.)
fail
0 / 1
test_huffman_properties_extra_improve_tree.py::test_improve_tree (Test that improve_tree does not modify its dictionary argument.)
pass
1 / 1
test_huffman_extensive_improve_tree.py::test_improve_tree (Test if improve_tree produces a tree equivalent to the instructor's.)
fail
0 / 1
•
Pyta compress.py
pass
10 / 10