Trinity Belen

Project 3 Analysis

**Pseudo-Code (on most important parts of the code)**

compress\_image(in\_file\_name, out\_file\_name):

image -> Image.open(in\_file\_name)

size\_raw -> raw\_size(image.height, image.width)

counts -> count\_symbols(image)

tree -> build\_tree(counts)

trimmed\_tree -> trim\_tree(tree)

codes -> assign\_codes(trimmed\_tree)

size\_estimate -> compressed\_size(counts, codes)

stream -> OutputBitStream(out\_file\_name)

encode\_header(image, stream)

stream.flush()

encode\_tree(trimmed\_tree, stream)

stream.flush()

encode\_pixels(image, codes, stream)

stream.close()

size\_real -> stream.bytes\_written

decompress\_image(in\_file\_name, out\_file\_name):

stream -> InputBitStream(in\_file\_name)

height, width -> decode\_header(stream)

stream.flush()

trimmed\_tree -> decode\_tree(stream)

stream.flush()

image -> decode\_pixels(height, width, trimmed\_tree, stream)

stream.close()

image.save(out\_file\_name)

**Time Complexity**

There are O(n) iterations, one for each item/node of the tree, thus the time complexity is:

O(n log n)