1. Everything is fine, luckily, or at least it seems fine

2.

MyHash.h

I used a dynamically allocated array of pointers to a linked list of node, which includes key, value, and next.

void associate(const KeyType& key, const ValueType& value)

if its not in there

put it in and return

if it doesn’t go over the max load factor

if that bucket has nothing linked to it

link a new node to it

if that bucket has something linked to it

go to the last node and link the new node there

if it goes over the max load factor

build a new array with twice the size

get everything out of the old array and put them in the new one

delete the old array

tokenizer

in private member, I used a vector of character to store the separators

wordlist

I used a myhash from string of pattern to a vector of string that fits the pattern

I used a helper function called makepattern

vector<string> WordListImpl::findCandidates(string cipherWord, string currTranslation) const

I make the pattern and then search in the myhash for everything that fits the pattern and fits the current translation

Translator

I used a linked list of nodes, which contains two myhash from char to char, used to make two maps of different directions

string TranslatorImpl::getTranslation(const string& ciphertext) const

look for the translation of each char from the map and add it to the output string

Decrypter

I used the exact algorithm in the spec

3. Luckily, I think it satisfies the big o requirements