CSE 15: Discrete Mathematics Laboratory 6

Spring 2019

Introduction

This lab is the first in a series of labs dedicated to cryptography. So far we have only seen the Caesar Cipher, so we will be implementing it here. Before we get to that, there is the matter of assigning numerical values to letters of the alphabet. In class, for the sake of simplicity, we mapped the English alphabet to the set {0..25}. This of course limits our ability to represent lower case letters, spaces, numbers, and punctuation symbols.

So there are more than 26 characters we need to be able to represent, but exactly how many is it? To answer this question, we look at the American Standard Code for Information Interchange, abbreviated as ASCII. If you are unfamiliar with ASCII codes, take a look at `https://en.wikipedia.org/wiki/ASCII`, but essentially it is an encoding scheme that assigns integer values to characters. There are 95 printable ASCII characters, and they are from 32 to 126. For example the character "A" has an ASCII value of 65, "B" is mapped to 66, and "a" is 97. The space character is 32. The Wikipedia article referenced earlier shows the entire table.

The Python language provides functions for converting characters to ASCII values and vice versa. To get the ASCII value of a character, use `ord(c)`. For example `ord('A')` returns 65, and `ord('a')` returns 97. There is also a way of converting an integer to its corresponding ASCII character. This is done with the `chr(n)` function. For example, `chr(98)` returns `'b'`.

So we will use the `ord()` and `chr()` functions to convert from letters to numbers and vice versa, which means that our alphabet has 127 characters, but the first 32 characters are not printable, so we need to exclude them. We will therefore convert a character to a number by taking its ASCII value and subtracting 32. That way we go from a range {32..126} to {0.. 94}. Follow a similar process when converting from a number back to a character. Our shift ciphers will then be using modulo 95.

Exercises

The file `caesar.py` contains incomplete implementations of the following functions. Your task is to complete them, and upload your `caesar.py` file to CatCourses.