Objectives

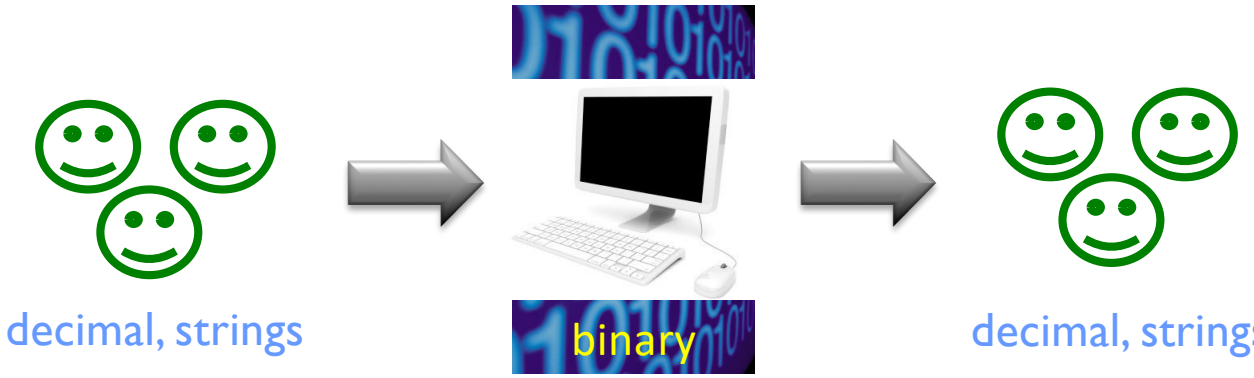
- More: computer's representations of data types
- Encryption

Review

- What is the special name for sequences, like newlines, tabs, …?
  - How do we represent them in strings?
- How does the computer represent data (e.g., numbers and text)?

Review: Representations of Data

- Computer needs to represent different types of data
  - Eventually, all boils down to 1s and 0s
- Computer needs to translate between what humans know to what computer knows and back again



Seems like a divergence on strings but just wait…

String Representations

- A **string** is a *sequence* of characters
- Each character is stored as a binary number
- **ASCII** (American Standard Code for Information Interchange) is one standard encoding for characters
  - Limitation: ASCII is based on the English language
  - Cannot represent other types of characters
  - Handout is just a subset
- Unicode is a new standard – handles all languages

Translating to/from ASCII

- Translate a character into its ASCII numeric code using **built-in function** `ord`
  - `ord('a') ==> 97`
- Translate an ASCII numeric code into its character using **built-in function** `chr`
  - `chr(97) ==> 'a'`

`ascii_table.py`
`ascii.py`

ASCII Questions

- Lowercase letters are represented by what range of numbers?
- Uppercase letters are represented by what range of numbers?
- What is the difference between the decimal encoding of ‘M’ and ‘N’?
  - Between ‘m’ and ‘n’?
- Explain why `"Zebra" < "aardvarks"` evaluates to `True`

ASCII Questions

- Lowercase letters are represented by what range of numbers?
  - 97—122
- Uppercase letters are represented by what range of numbers?
  - 65—90
- What is the difference between the decimal encoding of 'M' and 'N'?
  - Between 'm' and 'n'?
  - 1
- Explain why `"Zebra" < "aardvarks"` evaluates to `True`
  - `ord("Z") < ord("a")`

Translating to/from ASCII

- Translate a character into its ASCII numeric code using **built-in function** `ord`
  - `ord('a')` evaluates to `97`
- Translate an ASCII numeric code into its character using **built-in function** `chr`
  - `chr(97)` evaluates to `'a'`

`ascii_table.py`
`ascii.py`

Encryption

- Process of encoding information to keep it secure
- One technique: Substitution Cipher
  - Each character in message is replaced by a new character

Encryption: Caesar Cipher

- Replace character with a character X places away
  - X is called the ***key***
- Julius Caesar used technique to communicate with his generals

Original Letter	Key	Encrypted Letter
‘a’	1	‘b’
'b’	1	‘c’
‘z’	1	‘a’

- “Wrap around” within the lowercase letters
- Write program(s) to do this in next lab

Caesar Cipher

- Using the ASCII handout, what would be the encoded messages?

Message	Key	Encoded Message
apple	5	
zebra	5	
the eagle flies at midnight	-5	

Caesar Cipher

Message	Key	Encoded Message
apple	5	fuuqj
zebra	5	ejgwf
the eagle flies at midnight	-5	ocz zvbgz agdzn vo hdyidbco

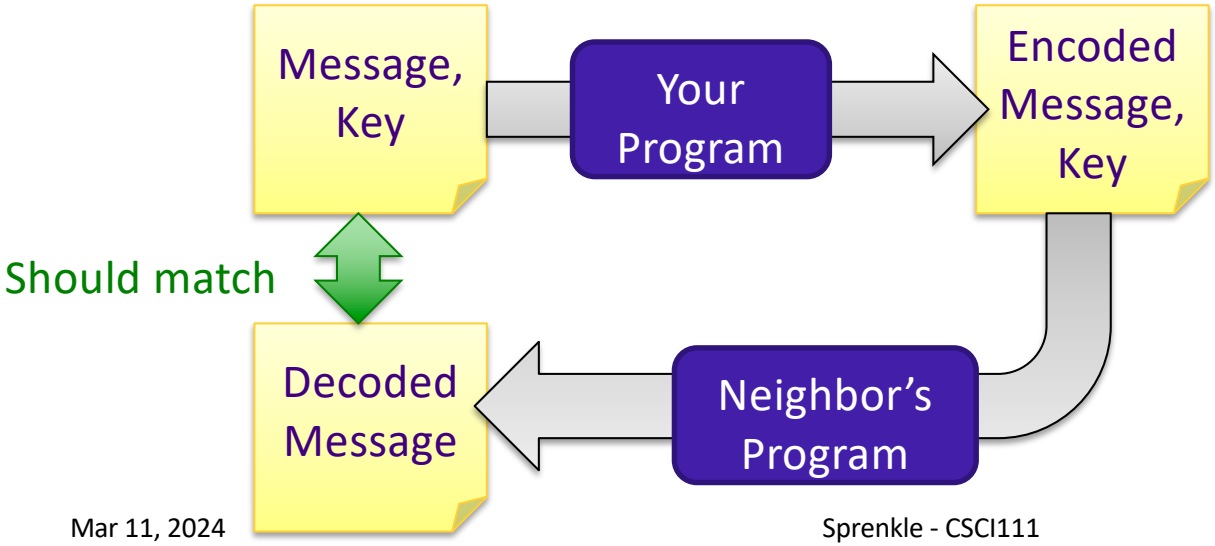
How would you *decode* an encrypted message?

Top-Down Design

- Break a problem into subproblems
  - Continue process until you reach “base problems” to solve

Next Lab

- Write an encoding/decoding program
  - Encode a message
  - Give to a friend to decode



What is your algorithm for the encoding process?
→ Break into pieces

Top-Down Design

1. Get user input for message and key
2. Check that the message and key are valid
3. Encrypt the message using the key
4. Output the encrypted message

Top-Down Design

1. Get user input for message and key
2. Check that the message and key are valid
3. Encrypt the message using the key
4. Output the encrypted message

Break this down: what happens in this step?

Top-Down Design: Encrypt Message

- Go through each character in the message and encrypt it

Top-Down Design: Encrypt Message

- Go through each character in the message and encrypt it

Encrypt Letter

- API: Takes a *lowercase letter* and a *key* and returns the encrypted letter
- Write test cases
- Write algorithm
- What are the preconditions for the function?

Caesar Cipher: `encryptLetter`

- Given a letter and key
- Convert the character to its ASCII value
- Add the key to that value
- Make sure that the new value is a "valid" ASCII value, i.e., that that new value is in the range of lowercase letter ASCII values
  - If not, "wrap around" to adjust that value so that it's in the valid range
- Convert the ASCII value into a character
- Return the encrypted letter

Top-Down Design: Encrypt Message

Original algorithm: Go through each character in the message and encrypt it

- Now that we have the encryptLetter function, consider the algorithm and implementation of the encryptMessage function
  - What are good test cases?
  - What are the preconditions for the function

Caesar Cipher (Partial) Algorithm

- Given a message and key
- For each character in the message
  - Check if the character is a lower case letter
    - If it is, encrypt it
    - Otherwise, it stays that character
- Return the message

Looking Ahead

- Pre Lab 7 due before lab
  - Shorter assignment
  - Some repetition with last week's assignment, as we go into more depth on some topics
- Think about the encoding/encryption problem and how you will implement it
- Broader Issue: cryptography