# LECTURE 29 **REGULAR EXPRESSIONS 2: ENCODINGS AND BINARY FILES**

MCS 260 Fall 2020 Emily Dumas

### REMINDERS

- I hope you have worked on Project 3
- Quiz 10 due Monday (Nov 2)
- Nov 3: No discussions
- Nov 5: Discussion converted to TA office hours

#### **REGEX QUICK REFERENCE**

- . matches any character except newline
- $\sames$  matches any whitespace character
- \d matches a decimal digit
- + previous item must repeat 1 or more times
- \* previous item must repeat 0 or more times
- ? previous item must repeat 0 or 1 times
- {n} previous item must appear n times
- (...) treat part of a pattern as a unit and capture its match into a group
- [...] match any one of a set of characters
- A | B match either pattern A or pattern B.
- ^ match the beginning of the string.
- \$ match the end of the string or the end of the line.

## **RE MODULE QUICK REFERENCE**

- re.search(pattern, string) does string
   contain a match to the pattern? Return a match
   object or None.
- re.finditer (pattern, string) Return an iterable containing all non-overlapping matches as match objects.
- re.findall (pattern, string) return a list of all non-overlapping matches as strings.

## **EXAMPLE PROBLEM**

Find all of the phone numbers in a string that are written in the format 319–555–1012, and split each one into area code (e.g. 319), exchange (e.g. 555), and line number (e.g. 1012).

### **SQUARE BRACKETS**

Give a list of characters and to match any one of them.
[abc] matches any of the characters a, b, c.
[^abc] matches any character except a, b, c.
[A-Za-z] matches any alphabet letter.
[0-9a-fA-F] matches any hex digit.



A | B matches either pattern A or pattern B.

Use this inside parentheses to limit how much of the pattern is considered to be part of A or B, e.g.

[Hh](ello|i),? my name is (.\*).

## **FINDING FUNCTIONS**

Let's make a program to find function definitions in a Python source file and print the function names.

## **ENCODING PREVIEW**

What is the size of a file if we open and write one of these words to it?

- Hello (5 characters)
- Frühstück (9 characters)
- 😊 (1 character, U+1F60A)

Note: The last item in the list above has an emoji which doesn't render correctly in the PDF slides.

## ENCODING

As the OS sees it, a file is a sequence of bytes. To write text, we need to decide how to represent code points as bytes.

A scheme to do this is an **encoding**. Encodings can also specify which code points are allowed.

The default encoding in Python is usually UTF-8, though officially this is platform-dependent.

In UTF-8, the first 128 code points are stored as a single byte. Others become two, three, or four bytes.

#### **BINARY FILES**

- **Opening a file with** "b" **in its mode string will make it a binary file.** E.g. "rb" reads a binary file, "wb" writes to one.
- Reading from a binary file gives a bytes object, a sequence of ints in the range 0 to 255.
- We can decode bytes into a string with the method .decode(), and can encode a string as bytes with .encode(). Each takes optional encoding parameter.

#### REFERENCES

- In Downey:
  - Regular expressions, character encoding, and binary files are not discussed.
- The official Python tutorial has a section about reading and writing files which discusses binary files and encoding.
- Pythex is a free online regular expression editor and tester that can be very helpful for debugging patterns.
- Google's free online Python course has a unit on regular expressions.
  - This course was developed for Python 2, so calls to print are lacking parentheses.
     Otherwise, the code should work.
- The documentation of the re module is good as a reference, but may not be ideal to learn from.

#### **REVISION HISTORY**

• 2020-10-29 Initial publication