Python Class 16

Saving data

Working Gene Pattern

```
def CountPattern(self,pattern):
    if not pattern[0] in self.codon_list:
        return 0
    else:
        count = 0
        for codon in self.codon_list:
            if codon == pattern[0] and self.codon_list.index(codon) + len(pattern) > len(self.codon_list):
                break
            elif codon == pattern[0]:
                index = self.codon_list.index(codon)
                test = []
                for x in range(len(pattern)):
                    test.append(self.codon_list[index+x])
                if test == pattern:
                    count +=1
            else:
                pass
        return count
```

Saving data

Open a file

- To open a file, we use the built-in function open()
- open() takes two arguments
  - *file*
  - *mode*

Open arguments

- open(***file***,mode)
  - file is the path and name of the file
  - **For example** *C:\Users\David Hunter\Documents\Seitoku Python Curriculum*
  - This is the path where I save my files for this class.
  - *C:\Users\David Hunter\Documents\Seitoku Python Curriculum/**2021 Python Class 11.odp***
  - The **BOLD** part is the file name.

Open arguments, 2

- open(file,***mode***)
  - mode is how you want to open the file
  - “r” -reads the file. Error if no file. (this is the default (初期設定))
  - “a” -appends. Creates the file if no file.
  - “w” - opens the file to write. Creates the file if no file
  - “x” - create the file. Error if the file exists.

Let's Practice

- Open/Make a file using the different modes – a,r,w,x

Mode （ファイルの読み方）

- You can also say how the file should be read.
  - “b” is for binary （二進法 (にしんほう)） → People cannot read this format, but computers can.
  - “t” is for text (this is the default (初期設定 (しょきせってい)))

File variables

- file.closed – True or False
- file.mode- the mode
- file.name – the name
- ~~file.softspace – not really important~~

File functions

- file.close() - closes the file
- file.read() - this gives you the data/information in the file
- file.write(str) – this adds a string to the file
- file.writelines(sequence) – this adds a list of strings to the file

Practice adding some information to a file

```
my_file = open("Hello.txt", "w")
print(my_file.name)
print(my_file.mode)
my_file.write("Hello")
my_file.close()
my_file = open("Hello.txt", "r")
print(my_file.read())
```

Practice reading information from a file

```
my_file = open(“Hello.txt”, “r”)
data = my_file.read()
```

Practice saving text

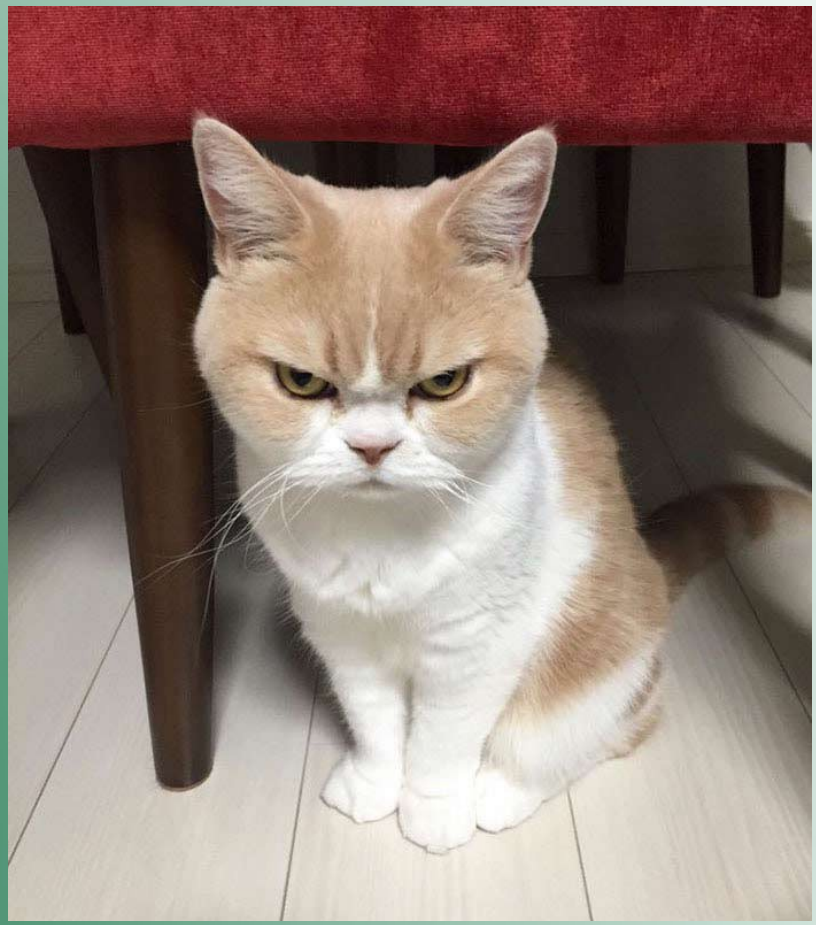
- Use the DNA class and save some data.

Using pickle to save data

- Pickle is a library, just like datetime or random.
- Pickle is used for saving data.
- It is especially useful for saving complicated data, such as class instances, lists, etc.

Why not file.write(str)?

- We can only use strings for file.write.
- Integers, floats, dates, bools, and classes that we make (Gene, DNA, To-DoItem....) need to be changed into strings to save, and back into the class to read



How to use

- Pickle has two main functions we will use
  - pickle.dump(obj,file)
    - obj is the information we want to save
    - file is a file that we opened
  - pickle.load(file)
    - this reads the data from the file and returns the data

Example

```
import pickle
import datetime

class Book:
    def __init__(self, n= "", a = "",d = datetime.date.today()):
        self.name = n
        self.author = a
        self.publish_date = d

    def __str__(self):
        return self.name + " by " + self.author + \
               " (" + str(self.publish_date.year) + ")"

book = Book("The Malazan Book of the Fallen", \
            "Steven Erikson", datetime.date(2001,1,1))


file = open("test","wb")
pickle.dump(book,file)

file.close()

infile = open("test","rb")

correct = pickle.load(infile)
infile.close()

print(correct)
```

Practice!!!

- Pick one of the projects we made in previous classes: recipe, manga-ka, vending machine, to-do list.
- Use pickle to save the data to a file.
- Use pickle to load the data.