Programming in Python[1]

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia
mattia.monga@unimi.it

Academic year 2020/21, II semester

Lecture VI: Dictionaries, sets, comprehensions

Homework status

- Students: 27
- One triangle: tried by 22, 16 correct solutions
- Triangle kinds: tried by 20, 9 correct solutions
- DNA Hamming: tried by 19, 12 correct solutions
- Newton Sqrt: tried by 17, 13 correct solutions
- 7 students did all the exercises correctly

Dictionaries

A composite type `dict` that implements a mapping between immutable keys and values.

```
d = {'key': 'foo',  3: 'bar'}

print(d['key']) # 'foo'
print(d[3])     # 'bar'
print(d[2])     # error!
```

Notation is similar to lists/tuples, but `dict`s are not sequences indexed by numbers, you must use only the existing keys (`d.keys()`).

```
if x in d.keys():
   print(d[x])
```

A sequence of values can be obtained with `d.values`. A sequence of 2-tuples `(key, value)` with `d.items()`.

Sets

A `set` is a composite object with no duplicate (non mutable) elements. Common set operations are possible.

- Set literals: `{1,2,3}` `set()`
- `{1,2,3}.union({3,5,6})`
  `{1,2,3}.intersection({3,5,6})`

Comprehensions

Comprehensions are a concise way to create lists, sets, maps... It resembles the mathematical notation used for sets $A = \{a^2 | a \in \mathbb{N}\}$.

```
squares = [x**2 for x in range(10)]

# equivalent to:
squares = []
for x in range(10):
  squares.append(x**2)

# filtering is possible
odds = [x for x in range(100) if x % 2 != 0]

# with a set
s = {x for x in range(50+1) if x % 5 == 0}

# with a dict
d = {x: x**2 for x in range(10)}
```


Files

A file is an abstraction the operating system uses to preserve data among the execution of programs. Data must be accessed **sequentially**.

- We need commands to ask to the OS to give access to a file (`open`).
- It is easy to read or write data **sequentially**, otherwise you need special commands (`seek`) to move the file "cursor"
- The number of open files is limited ($\approx$ thousands), thus it is better to `close` files when they are not in use

Files contain bits (normally considered by group of bytes, 8 bits), the interpretation ("format") is given by the programs which manipulate them. However, "lines of printable characters" (plain text) is a rather universal/predefined interpretation, normally the easiest to program.

File read access


```
f = open('filename.txt', 'r') # read only

# iterating on a file reads (all) the lines
for i in f:
    print(i)

# End of file already reached, result is ''
f.readline()

f.close()

# File closed, error!
f.readline()
```

To avoid remembering to close explicitly, Python provides the context manager syntax.

```
with open('filename.txt', 'r') as f:
  for i in f:
     print(i)
```

Exercises

- Write a function to compute the complement of a DNA strand: every `A` becomes a `T`, every `T` an `A`, every `C` an `G`, every `G` an `C`.
- Apply the function to every line of a file with a DNA sequence
- Write a function that gives the set of (unique) sequences of 10 nucleic acids in a file

Homework

- `https://classroom.github.com/a/MhchQHAd`
- `https://classroom.github.com/a/36ITXw1V`