Lecture 13: Lists

CS 51P

October 23, 2023

Programs operate on values

- compute new values using expressions
- store values in variables
- pass values to functions (as arguments)
- return values to caller (as return value)

Can we operate on multiple values at the same time?

- Store colors of the rainbow?

  color1 = “red”
  color2 = “orange”
  color3 = “yellow”
  color4 = “green”
  color5 = “blue”
  color6 = “purple”
  Not efficient!!

- Can we define a variable that stores the colors of the rainbow?

- Can we define a function that returns the colors of rainbow in uppercase?

Data Structures

- a **data structure** is a type that stores a collection of values
- Python provides some built-in data structure types

Sequences

- sequences are ordered sets of values
  - ranges are sequences of integers
  - strings are sequences of characters
  - files are sequences of strings

- we can perform operations on sequences
  - indexing (e.g., "hello"[0])
  - slicing (e.g., "hello"[1:3])
  - looping (with for loop) (e.g., for i in range(1,10):)
  - check membership (e.g., char in "abcd")

List is a sequence of arbitrary values

What is a List?

- a list is a sequence or an *ordered collection* of arbitrary items
  - Lists start/end with brackets with elements separated by commas.

```
a_list = [3, 6, 2, 1]
float_list = [5.1, 6.2, 0.23]
str_list = [‘this’, ‘is’, ‘a’, ‘list’]
mix_list = [3, 5.1, ‘is’, True]
empty_list = []
```

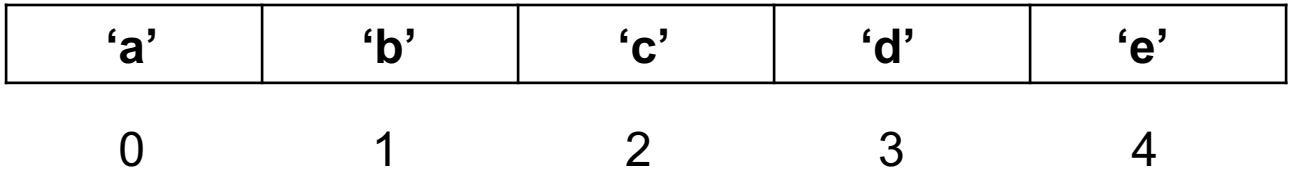
- a list is a sequence, so can index into, loop over, check for membership, slice
  - Lots of built-in functionality

Accessing elements of a list

- Consider this list:

```
a_list = ['a', 'b', 'c', 'd', 'e']
```

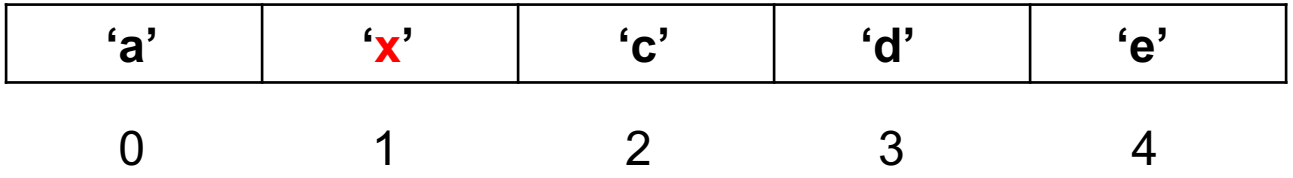
- a_list



- Accessing individual elements:
  - a_list[0] is 'a'
  - a_list[3] is 'd'

Modify elements of a list

- Consider this list: `a_list = ['a', 'b', 'c', 'd', 'e']`
- Can think of it like a series of variables that are indexed
  - E.g., a_list[0], a_list[1], a_list[2], …
- a_list



- Can modify individual elements like variables
  - a_list[1] = 'x'

Create list from string

- split() can be used to create a list from a string
  - split() will split the string based on whitespaces

```
s = “hello there!”
l = s.split()
print(l)
```

  - split(c) will split the string based on the given character c

```
s = “3,4,5,6”
l = s.split(“,”)
print(l)
```

Lists as sequences

```
string = "Hello world !! "
print(string[1:3])
print(string[-1])
print(string[:2])

str_list = string.split()
print(str_list)
print(str_list[1:3])
print(str_list[-1])
print(str_list[:2])
```

Example: Lists as sequences

- Can we define a variable that stores the colors of the rainbow?
  - colors = ["red", "orange", "yellow", "green", "blue", "purple"]

Differences about Lists

- the elements of a list can have any value and any type

```
a_list = [3.5, 6, [1, 2], "abc"]
```

- lists are mutable
  - add elements

```
a_list.append("c")
a_list.extend(["c","b"])
```

  - modify elements

```
a_list[3] = 3.33333
a_list[:2] = ["a", "b"]
```

  - remove elements

```
a_list.pop()
del(a_list[0:1])
```

List Operations

adding to a list (updates original list)

- a_list.extend(*list*)
- a_list.append(*elem*)
  - Different than extend – e.g. [5, 1]
- a_list.insert(*index*, *elem*)
  - Insert elem at index, shifts down

modifying a list

- direct assignment
  - a_list[0] = 2

removing from a list

- a_list.remove(*elem*)
  - removes 1st instance of elem
  - error if *elem* not in a_list
- del a_list[*slice*])
  - removes the slice from the list based on the given index
- a_list.pop()
  - returns (and removes) a_list[-1]
- a_list.pop(*index*)
  - returns (and removes) a_list[index]

Exercise

```
a_list = [3.5, 6, [1, 2], "abc"]
a_list[3] = list(range(0,5,2))
a_list[:2] = ["a", "b"]
a_list.extend([5,3,1])

print(len(a_list))
for elem in a_list:
     print(str(elem) + ":" + str(type(elem)))

del(a_list[3:5])
a_list.remove("a")
print(a_list)
```

Two ways to process each item in a list

- 1. iterate over items

```
for elem in l:
    print(elem)
```

- 2. iterate based on index

```
for i in range(len(l)):
    print(l[i])
```

Example

- Can we define a function that takes in a list of rainbow colors as input, and returns a list of rainbow colors with uppercase letters?

Exercise

- Define a function get_digits that takes one parameter num (an positive int) and returns a list of the digits of num