

PYTHON LISTS, TUPLES and METHODS

CSC 109 - Introduction to programming in Python - Summer 2023

Marcus Birkenkrahe

June 6, 2023

Contents

1	Overview	1
2	The list data type	3
3	Practice list creation and extraction	4
4	Working with lists	5
5	Augmented assignment operators	5
6	Methods	5
7	Example program: Magic 8 ball with a list	5
8	Sequence data types	5
9	References	5
10	Short program: Conway's Game of Life	5
11	Summary	5

1 Overview

- The `list` data type
- The `tuple` data type



Figure 1: Llyfrgell Genedlaethol Cymru / Llanfachraeth in darkness (1957)

- Augmented assignment operators
- Methods as type-specific functions
- References and pointers
- Conway's Game of Life

2 The list data type

- A **list** contains multiple values in an ordered sequence.
- A **list** is a *value* and can be stored in an object, and it also contains values also called *items*.
- The list items can be of any data type including lists:

```
print([1,2,3])    # numeric list (numeric items)
print(['cat','bat','rat','elephant'])  # string list (string items)
print(['hello', True, None, 42, 3.1415]) # mixed type list
```

```
[1, 2, 3]
['cat', 'bat', 'rat', 'elephant']
['hello', True, None, 42, 3.1415]
```

- Lists can be stored like any other value:

```
spam = ['cat', 'bat', 'rat']
print(len(spam))    # number of items in spam
print(type(spam))   # class of spam
print([] == list('')) # empty list
```

```
3
<class 'list'>
True
```

- **spam** is four things:
 1. a **list** variable (storage)
 2. a **list** value (stored)
 3. an ordered sequence of string values (indexed)
 4. an object (instanced)

```
spam = ["cat", "bat", "rat", "elephant"]
```

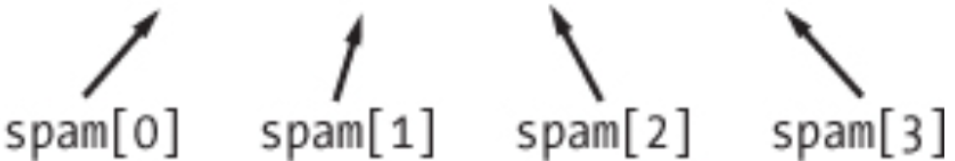


Figure 2: A list with its index values

3 Practice list creation and extraction

1. Assign these items to **spam** and extract them using a ranged for loop on one line separated by a single space: `cat bat rat elephant`

```
spam = ['cat', 'bat', 'rat', 'elephant']
for i in range(4):
    print(spam[i], end=' ')
```

```
cat bat rat elephant
```

2. What if the list has N elements? Can you generalize the loop?

```
for i in range(len(spam)):
    print(spam[i], end=' ')
```

3. Use elements of **spam** to print the sentence 'The bat ate the cat.' formatted with an f-string:

```
print(f"The {spam[1]} ate the {spam[0]}.")
```

4. Which error do you get when you use an index that exceeds the number of values in your list value? Create an example.

```
print(spam[5])
```

5. Can index values be non-integer? Find out!

```
print
```

```
def foo():  
    return 1 / 0 # This will cause a ZeroDivisionError  
  
foo()
```

- 4 Working with lists
- 5 Augmented assignment operators
- 6 Methods
- 7 Example program: Magic 8 ball with a list
- 8 Sequence data types
- 9 References
- 10 Short program: Conway's Game of Life
- 11 Summary