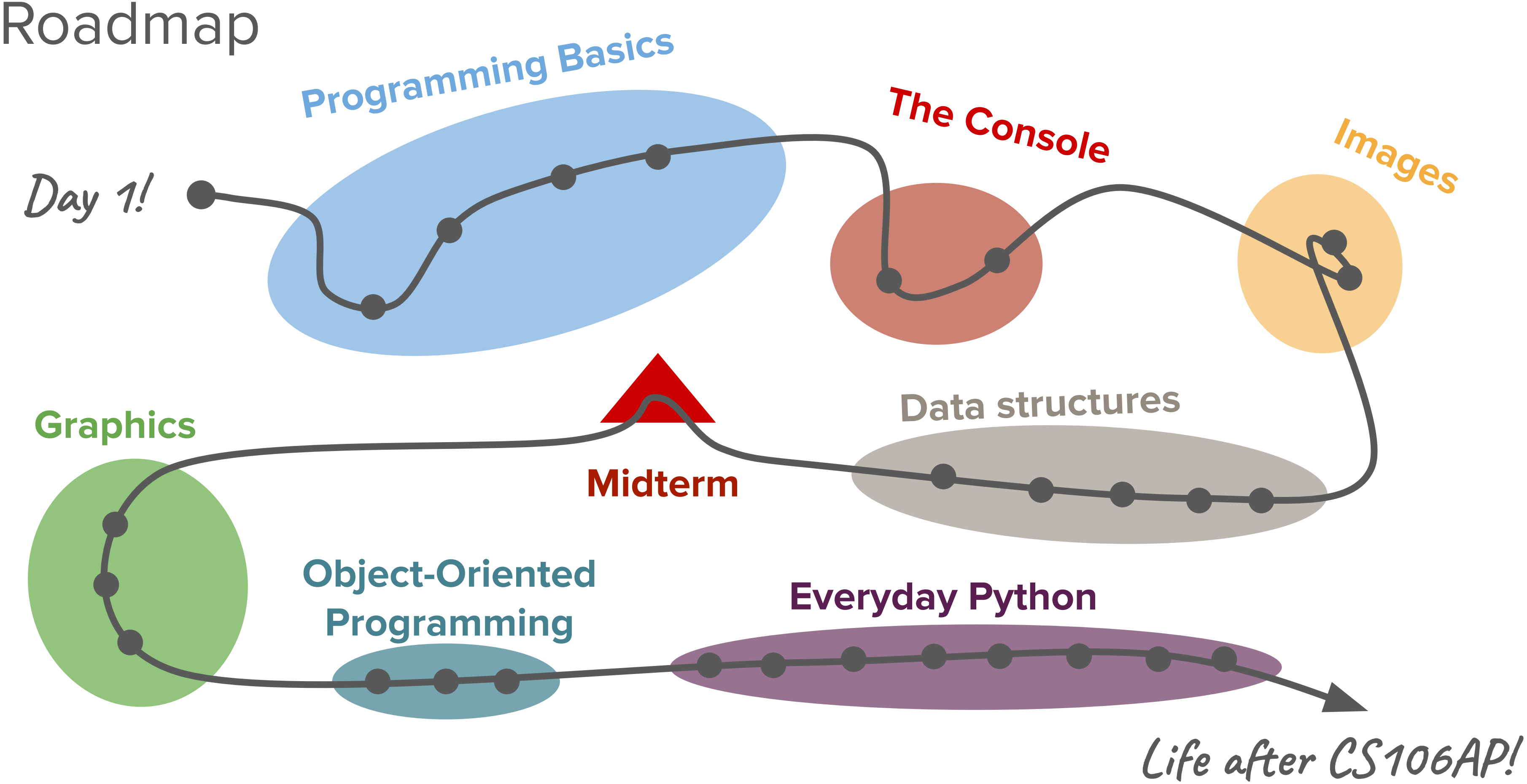
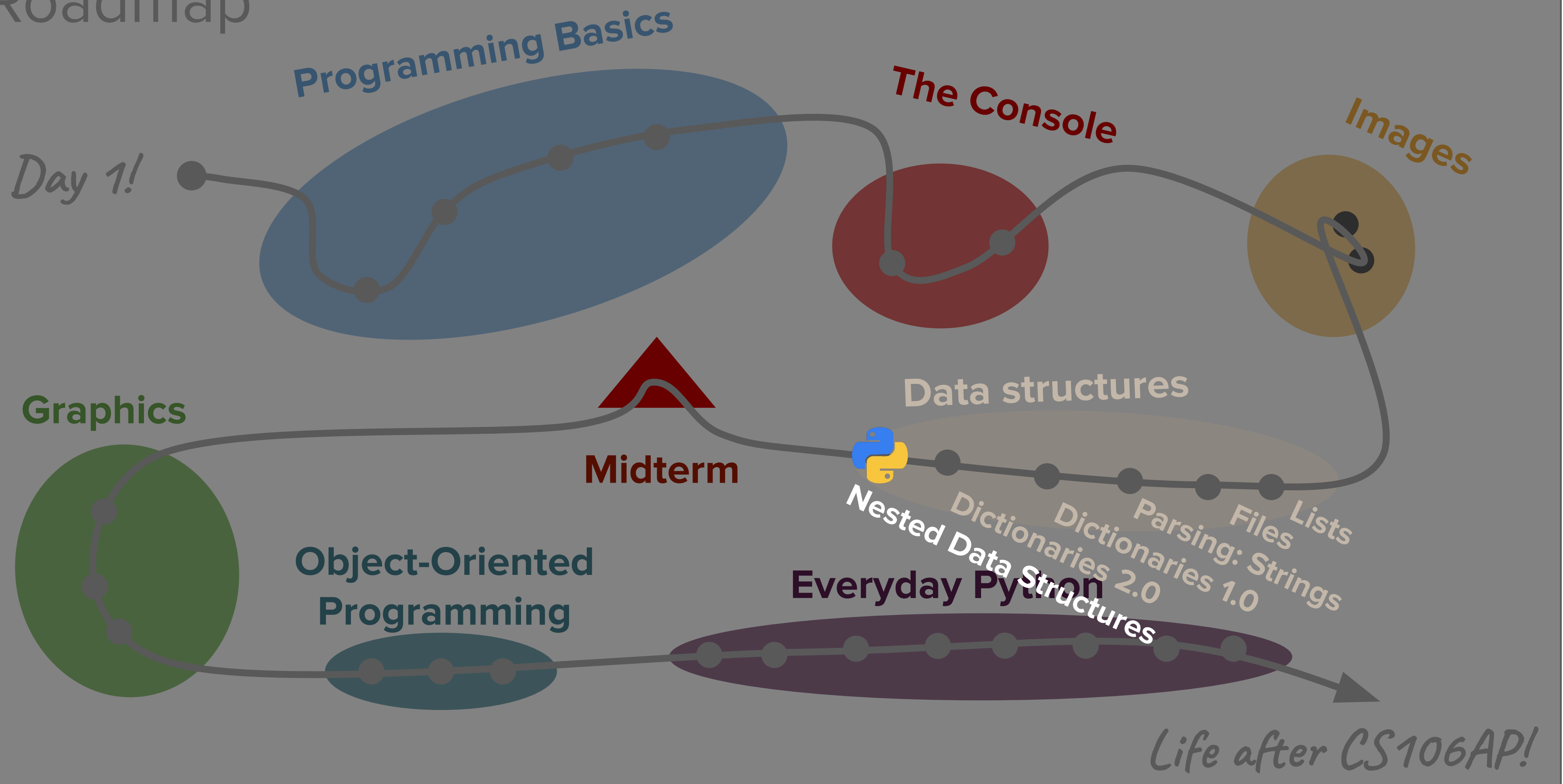
# Nested Data Structures

CS106AP Lecture 15

# Roadmap


Day 1!
Programming Basics
The Console
Images
Data structures
Midterm
Graphics
Object-Oriented Programming
Everyday Python
Life after CS106AP!

# Roadmap


Programming Basics
The Console
Images
Day 1!
Data structures
Graphics
Midterm
Lists
Files
Parsing: Strings
Dictionaries 1.0
Dictionaries 2.0
Nested Data Structures
Object-Oriented Programming
Everyday Python
Life after CS106AP!

# Today's questions

How can we store more information and add more structure to our data?

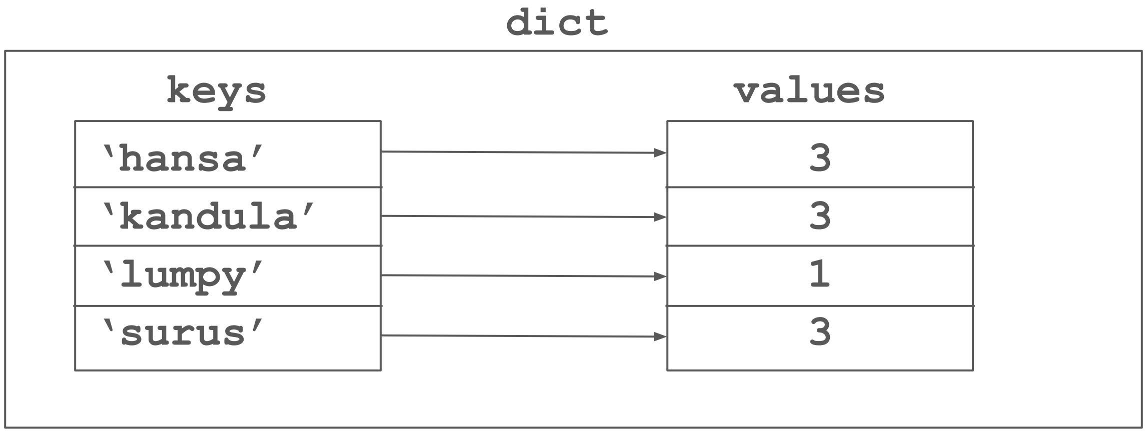
# Today’s topics

1. Review
2. Built-ins
3. Nested data structures
    - Lists
    - Dictionaries
4. What’s next?

# Review

# Big Picture: Dictionaries + Uniqueness

- A key will only be associated with one value
  - no duplicate keys!
- A dictionary can have multiple values that are the same.



dict

| keys | values |
|---|---|
| ‘hansa’ | 3 |
| ‘kandula’ | 3 |
| ‘lumpy’ | 1 |
| ‘surus’ | 3 |

# Accessing a Dictionary’s Keys

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}

>>> d.keys()

dict_keys(['Gates', 'MemChu', 'Tresidder'])
```

iterable collection of all the keys.

iterable means it can be used in foreach

# Accessing a Dictionary’s Keys

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}

>>> list(d.keys())

[‘Gates’, ‘MemChu’, ‘Tresidder’]
```

we are using list() to convert d.keys() into a list

# Accessing a Dictionary’s Values

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}

>>> list(d.values())

[23, 116, 57]
```

we are using list() to convert d.values() into a list

# Looping over a Dictionary’s Keys

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}
>>> for building in d.keys():
...       print(building)
Gates
MemChu
Tresidder
```

we can use foreach on the dictionary’s keys!

# Looping over a Dictionary’s Values

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}
>>> for age in d.values():
...     print(age)
23
116
57
```

we can use foreach on the dictionary’s values!

# Looping over a Dictionary’s Keys and Values

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}
>>> for building, age in d.items():
...     print(building, ‘is’, age, ‘years old.’)
Gates is 23 years old.
MemChu is 116 years old.
Tresidder is 57 years old.
```

items() gives us key, value pairs

# Printing with sep=

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}
>>> for building, age in d.items():
...       print(building, age, sep=‘: ’)
Gates: 23
MemChu: 116
Tresidder: 57
```

sep is an optional argument like end!

# Printing with sep=

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}
>>> for building, age in d.items():
...     print(building, age, sep=‘: ’)
Gates: 23
MemChu: 116
Tresidder: 57
```

the separating string will be printed between the arguments you pass into print()

# Getting a Sorted List of Keys

```
>>> d = {‘Gates’: 23, ‘Tresidder’: 57, ‘MemChu’: 116}

>>> sorted(d.keys())

['Gates', 'MemChu', 'Tresidder']
```

*sorted() returns a list in alphabetical order!*

# Retrieving Min/Max Values

```
>>> d = {‘Gates’: 23, ‘MemChu’: 116, ‘Tresidder’: 57}

>>> min(d.values())

23

>>> max(d.values())

116
```

returns the smallest element!

returns the biggest element!

## Definition

**Built-in Function**

A function built into Python that is always available for use.

# Examples of Built-ins

```
print()
input()
str()
int()
float()
len()
```

```
open()
list()
sorted()
max()
min()
```

# Built-ins with Lists

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
```

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst)
```

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst)
```

Creates an increasing sorted list

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst)
[-2, 5, 10, 34, 46]
```

Creates an increasing sorted list

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst)
[-2, 5, 10, 34, 46]
>>> lst
```

Creates an increasing sorted list

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst)
[-2, 5, 10, 34, 46]
>>> lst
[10, -2, 34, 46, 5]
```

Creates an increasing sorted list

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
```

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst, reverse=True)
```

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst, reverse=True)
[46, 34, 10, 5, -2]
```

# Sorted() in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> sorted(lst, reverse=True)
[46, 34, 10, 5, -2]
```

You can pass in an
optional parameter,
reverse=True.

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
```

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> max(lst)
```

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]

>>> max(lst)
```

Returns the maximum element in the list

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> max(lst)
46
```

*Returns the maximum element in the list*

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> max(lst)
46
>>> min(lst)
```

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> max(lst)
46
>>> min(lst)
```

*Returns the minimum element in the list*

# Max/Min in Lists

```
>>> lst = [10, -2, 34, 46, 5]
>>> max(lst)
46
>>> min(lst)
-2
```

Returns the minimum element in the list

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
```

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
```

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
```

We can use max/min on strings because characters have unicode representations

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
```

We can use max/min on strings because characters have unicode representations

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
```

We can use max/min on strings because characters have unicode representations

‘\u0064’, or 100 in decimal

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
>>> min(lst)
```

We can use max/min on strings because characters have unicode representations

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
>>> min(lst)
‘a’
```

We can use max/min on strings because characters have unicode representations

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
>>> min(lst)
‘a’
```

We can use max/min on strings because characters have unicode representations

‘\u0061’, or 97 in decimal

# Max/Min in Lists

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> max(lst)
‘d’
>>> min(lst)
‘a’
```

We can use max/min on anything where “<” has meaning.

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
```

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
```

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
```

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
```

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
```

extend() behaves like +=

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
>>> lst += [‘g’, ‘h’]
```

*extend() behaves like +=*

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
>>> lst += [‘g’, ‘h’]
>>> lst
```

extend() behaves like +=

# Extending a List

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst.extend([‘e’, ‘f’])
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’]
>>> lst += [‘g’, ‘h’]
>>> lst
[‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘g’, ‘h’]
```

extend() behaves like +=

# Note on Efficiency

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst += [‘e’, ‘f’]


>>> lst = lst + [‘e’, ‘f’]
```

# Note on Efficiency

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst += [‘e’, ‘f’]


>>> lst = lst + [‘e’, ‘f’]
```

This creates a new list
every time, so when the list
gets long, it’s inefficient.

# Note on Efficiency

```
>>> lst = [‘a’, ‘b’, ‘c’, ‘d’]
>>> lst += [‘e’, ‘f’]
```

This modifies in-place, so it’s fast!

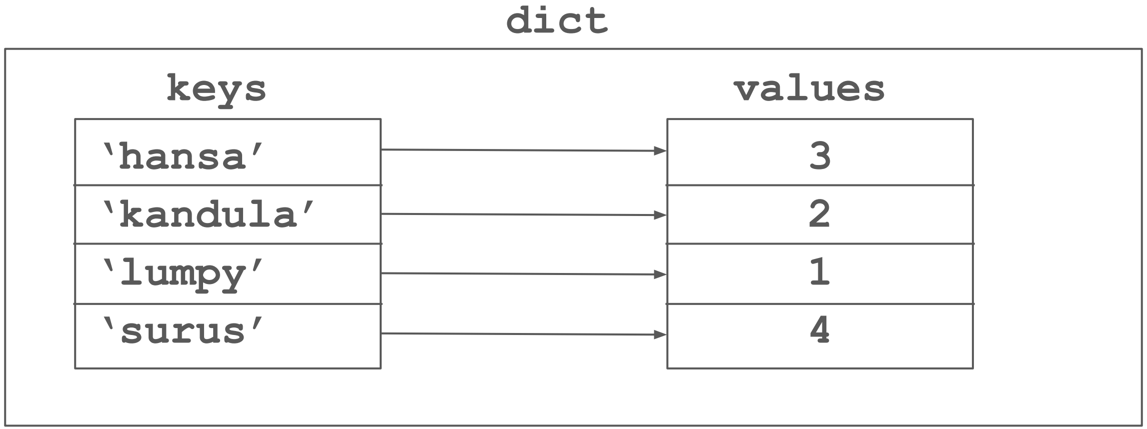
```
>>> lst = lst + [‘e’, ‘f’]
```

This creates a new list every time, so when the list gets long, it’s inefficient.

How can we store more information by adding more structure to our data?

# Recall: Animal – Feedings Dictionary

- animal name ⟶ number of feedings
- string ⟶ int

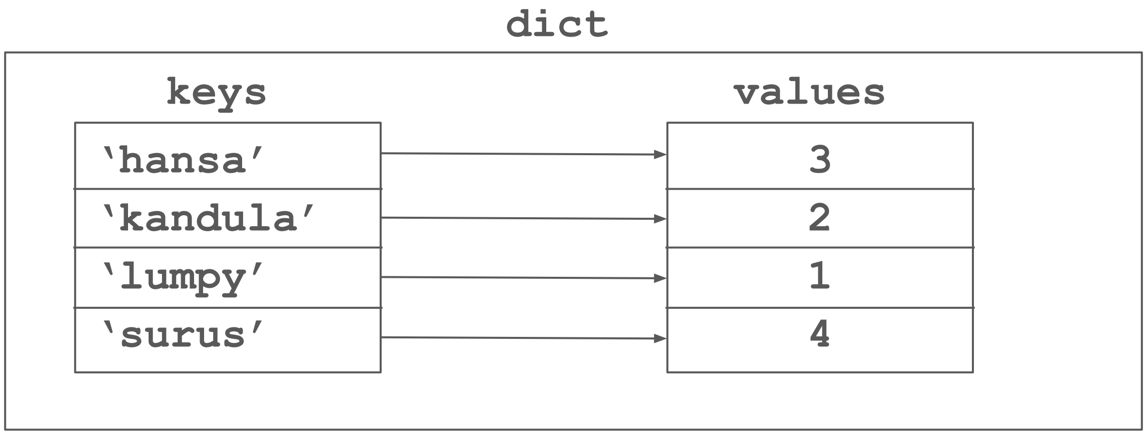


**dict**

| keys | values |
|---|---|
| ‘hansa’ | 3 |
| ‘kandula’ | 2 |
| ‘lumpy’ | 1 |
| ‘surus’ | 4 |

# Recall: Animal – Feedings Dictionary

- animal name ⟶ number of feedings
- string ⟶ int



`dict`

| keys | values |
|---|---|
| 'hansa' | 3 |
| 'kandula' | 2 |
| 'lumpy' | 1 |
| 'surus' | 4 |

*What if we wanted to store the* ***times*** *that the animals were fed?*

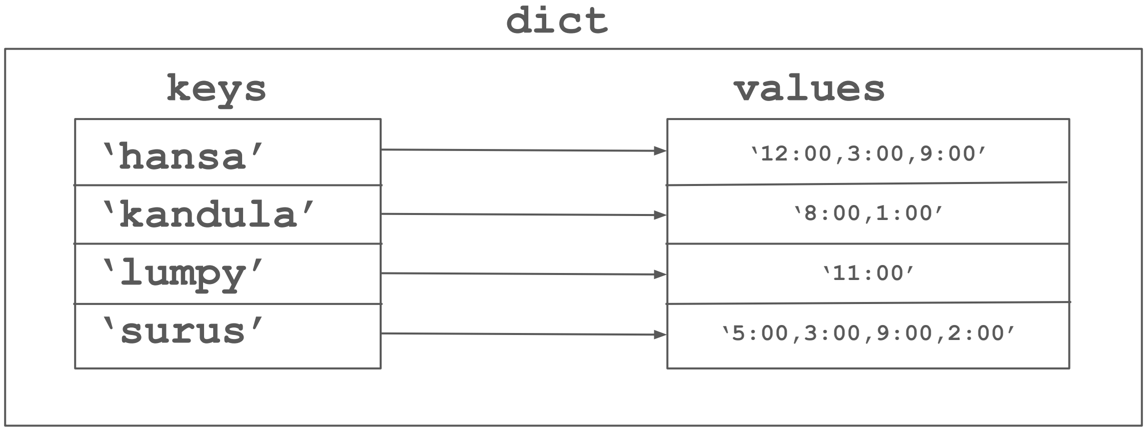
# Attempt #1: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **string**

*What if we wanted to store the* ***times*** *that the animals were fed?*

# Attempt #1: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **string**



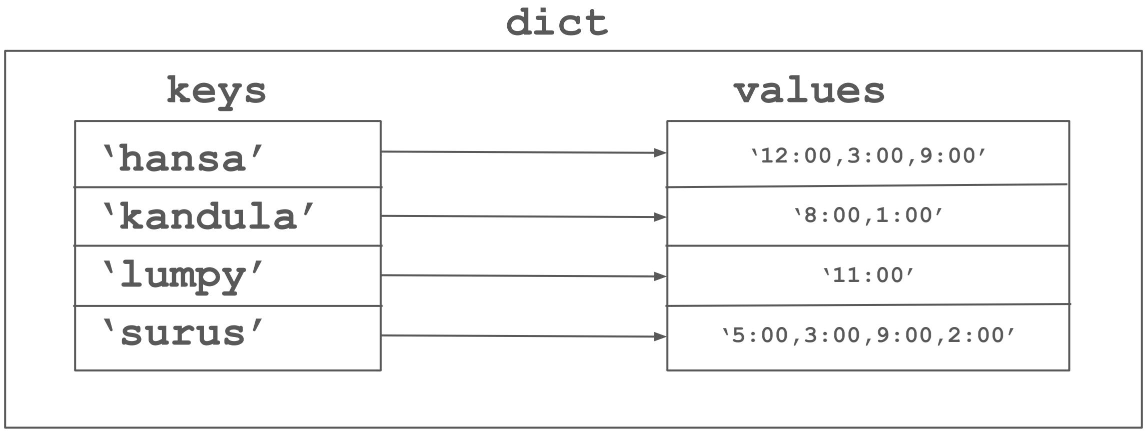
dict

| keys | values |
|---|---|
| ‘hansa’ | ‘12:00,3:00,9:00’ |
| ‘kandula’ | ‘8:00,1:00’ |
| ‘lumpy’ | ‘11:00’ |
| ‘surus’ | ‘5:00,3:00,9:00,2:00’ |

*What if we wanted to store the **times** that the animals were fed?*

# Attempt #1: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **string**



dict

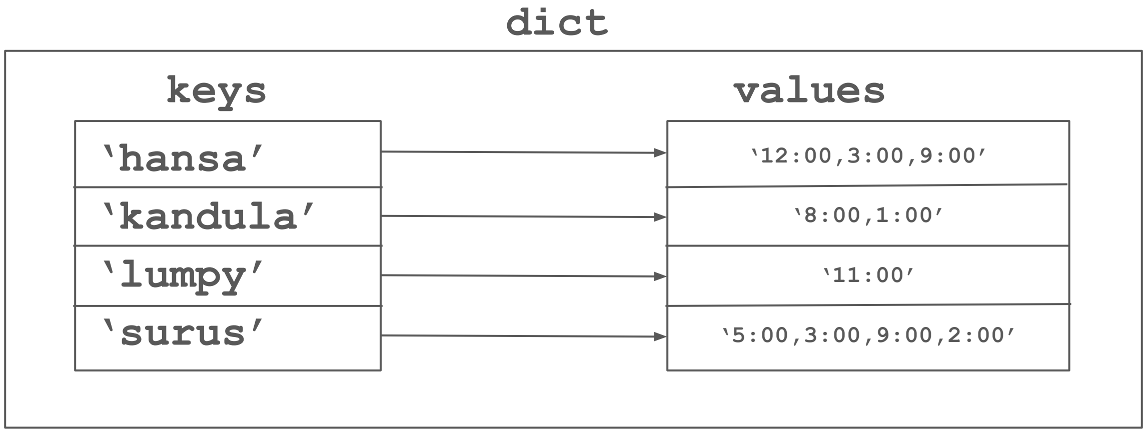
| keys | values |
|---|---|
| 'hansa' | '12:00,3:00,9:00' |
| 'kandula' | '8:00,1:00' |
| 'lumpy' | '11:00' |
| 'surus' | '5:00,3:00,9:00,2:00' |

What if we wanted to store the **times** that the animals were fed?

✗ Times are not easily accessible!

# Attempt #1: Animal – Feeding Times Dictionary

- animal name → **feeding times**
- string → **string**



dict

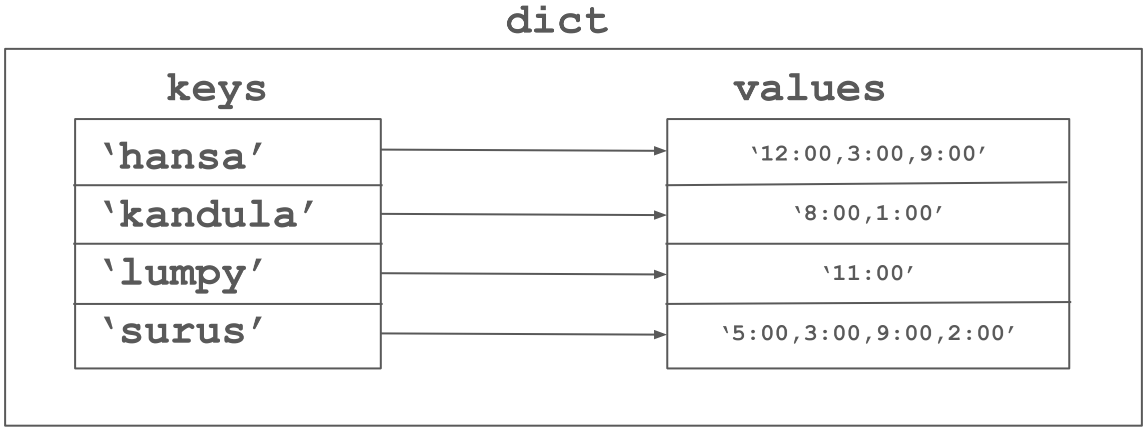
| keys | values |
|---|---|
| ‘hansa’ | ‘12:00,3:00,9:00’ |
| ‘kandula’ | ‘8:00,1:00’ |
| ‘lumpy’ | ‘11:00’ |
| ‘surus’ | ‘5:00,3:00,9:00,2:00’ |

What if we wanted to store the **times** that the animals were fed?

We’d have to call s.split(‘,’) anytime we wanted to access a time!

# Attempt #1: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **string**



dict

| keys | values |
|---|---|
| ‘hansa’ | ‘12:00,3:00,9:00’ |
| ‘kandula’ | ‘8:00,1:00’ |
| ‘lumpy’ | ‘11:00’ |
| ‘surus’ | ‘5:00,3:00,9:00,2:00’ |

What if we wanted to store the **times** that the animals were fed?

But those times look like a data type we know of.....

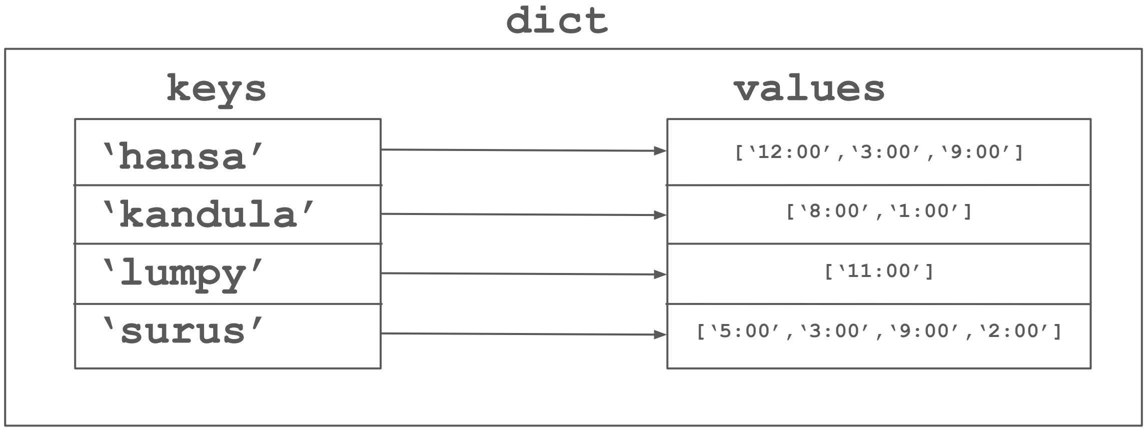
# Attempt #2: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **list[string]**

*What if we wanted to store the* ***times*** *that the animals were fed?*

# Attempt #2: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **list[string]**



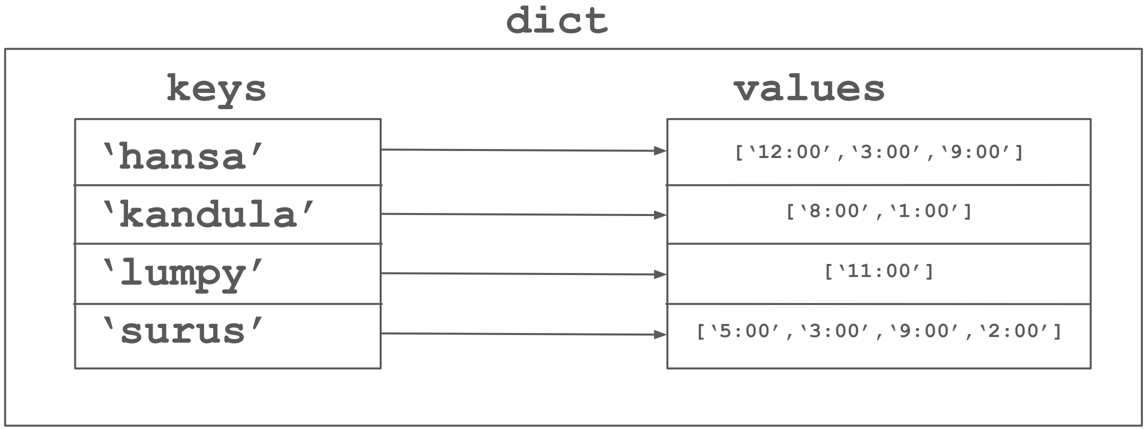
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

What if we wanted to store the **times** that the animals were fed?

# Attempt #2: Animal – Feeding Times Dictionary

- animal name ⟶ **feeding times**
- string ⟶ **list[string]**



dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

What if we wanted to store the **times** that the animals were fed?

We can easily access the individual times!

# Nested Data Structures

- We can nest data structures!

# Nested Data Structures

- We can nest data structures!
  - Lists in lists

# Nested Data Structures

- We can nest data structures!
  - Lists in lists
    - *grid/game board*

# Nested Data Structures

- We can nest data structures!
    - Lists in lists
        - *grid/game board*
    - Lists in dicts

# Nested Data Structures

- We can nest data structures!
  - Lists in lists
    - *grid/game board*
  - Lists in dicts
    - *animals to feeding times*

# Nested Data Structures

- We can nest data structures!
  - Lists in lists
    - *grid/game board*
  - Lists in dicts — (assignment 4)
    - *animals to feeding times*
  - Dicts in dicts

# Nested Data Structures

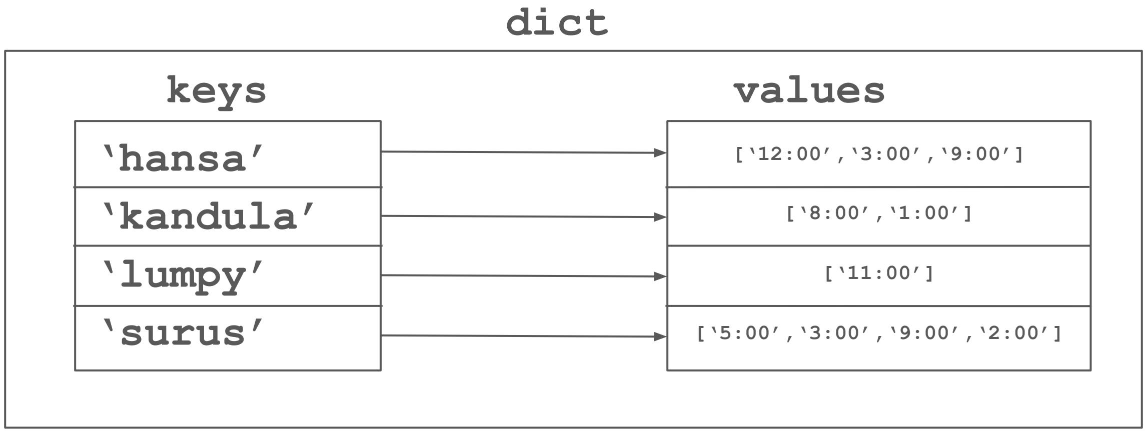
- We can nest data structures!
    - Lists in lists
        - *grid/game board*
    - Lists in dicts
        - *animals to feeding times*
    - Dicts in dicts
        - *your phone's contact book*

# Nested Data Structures

- We can nest data structures!
  - Lists in lists
    - *grid/game board*
  - Lists in dicts
    - *animals to feeding times*
  - Dicts in dicts
    - *your phone's contact book*
  - … and so on!

# Attempt #2: Animal – Feeding Times Dictionary

- animal name ⟶ **number of feedings**
- string ⟶ **list[string]**



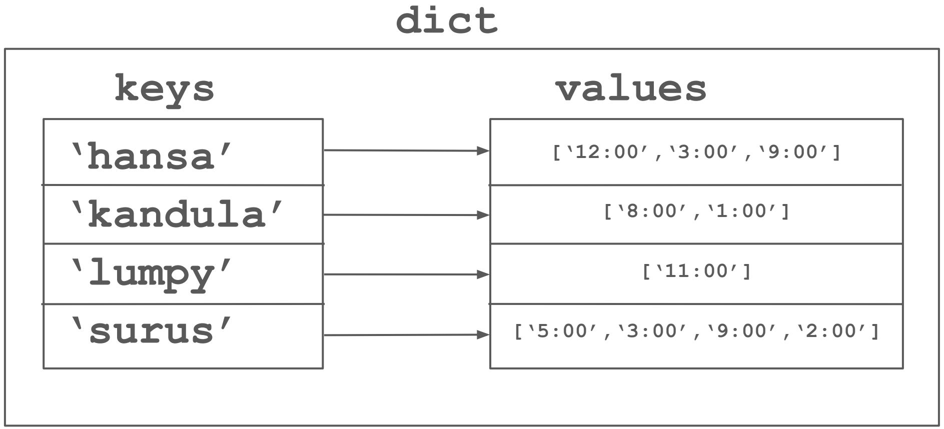
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

What if we wanted to store the **times** that the animals were fed?

How do we use this dictionary?

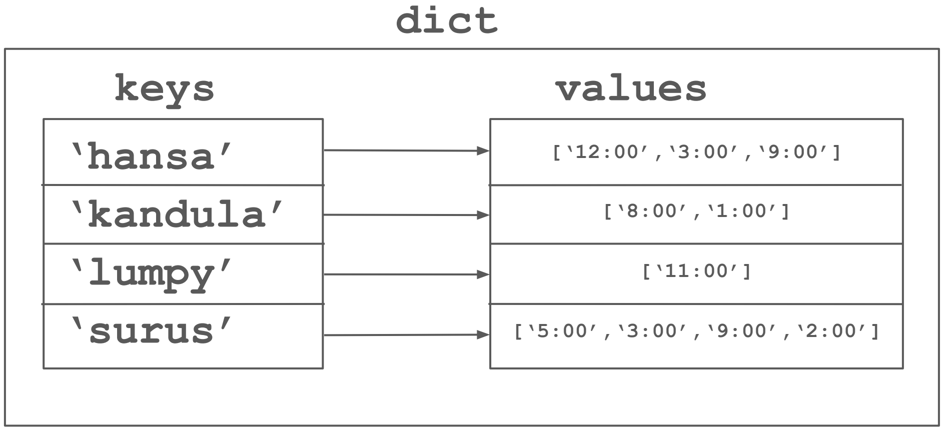
# Using a Dictionary Containing a List - Get



*Get the feeding times associated with ‘hansa’!*

# Using a Dictionary Containing a List - Get

```
>>> d[‘hansa’]
```



dict

keys

values

‘hansa’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’ → [‘8:00’,‘1:00’]

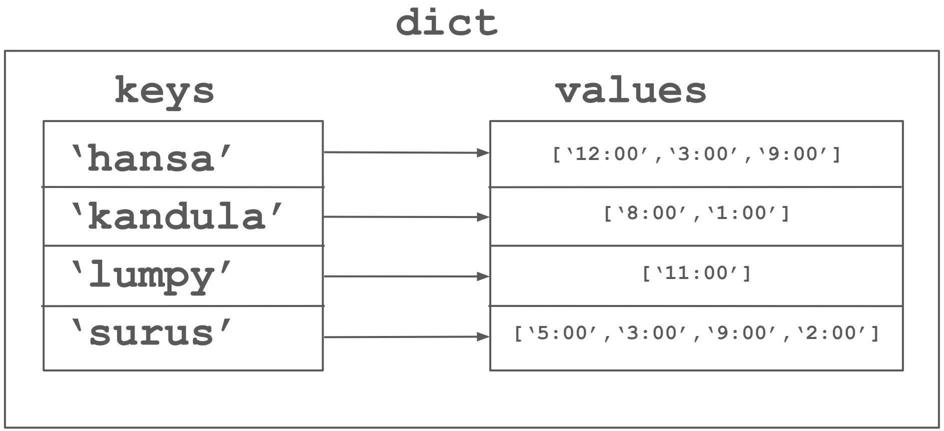
‘lumpy’ → [‘11:00’]

‘surus’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

*Get the feeding times associated with ‘hansa’!*

# Using a Dictionary Containing a List - Get

```
>>> d['hansa']
['12:00','3:00','9:00']
```



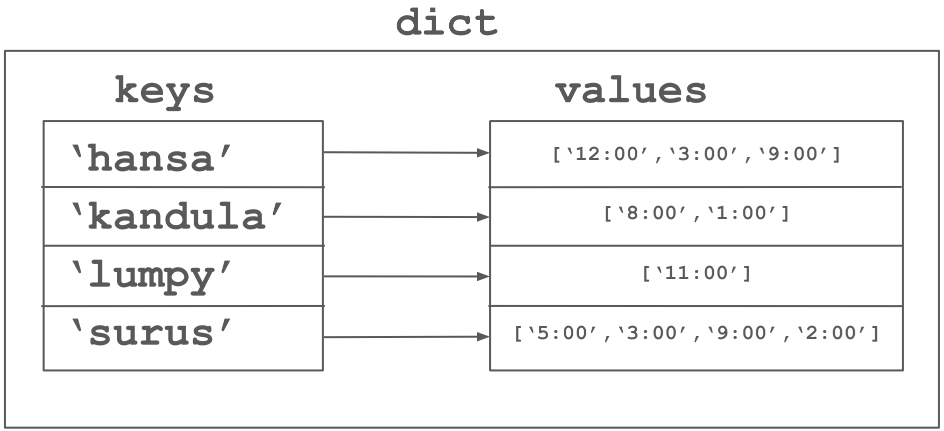
dict

| keys | values |
|---|---|
| 'hansa' | ['12:00','3:00','9:00'] |
| 'kandula' | ['8:00','1:00'] |
| 'lumpy' | ['11:00'] |
| 'surus' | ['5:00','3:00','9:00','2:00'] |

*Get the feeding times associated with 'hansa'!*

# Using a Dictionary Containing a List - Modify Value

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
```



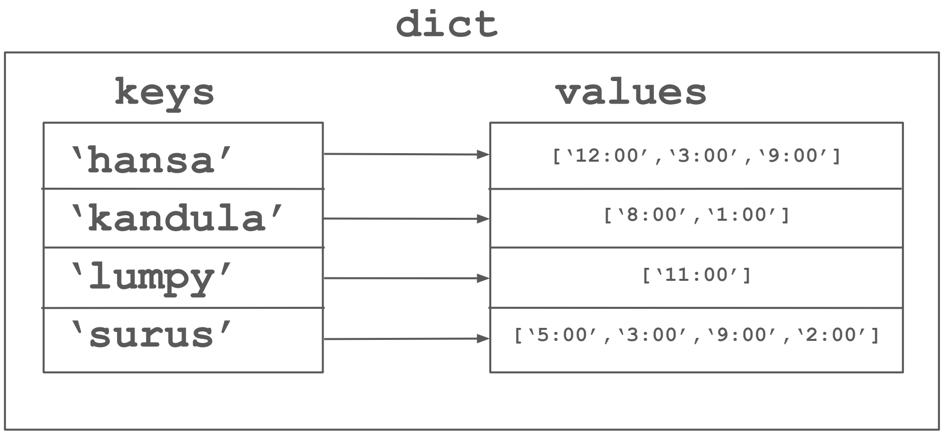
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Add a feeding time (‘4:00’) to ‘lumpy’!*

# Using a Dictionary Containing a List - Modify Value

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
```



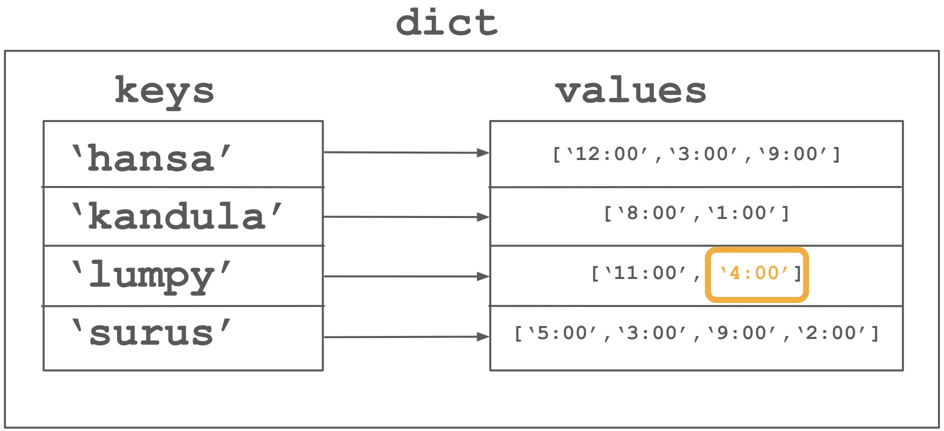
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Add a feeding time (‘4:00’) to ‘lumpy’!*

# Using a Dictionary Containing a List - Modify Value

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
```



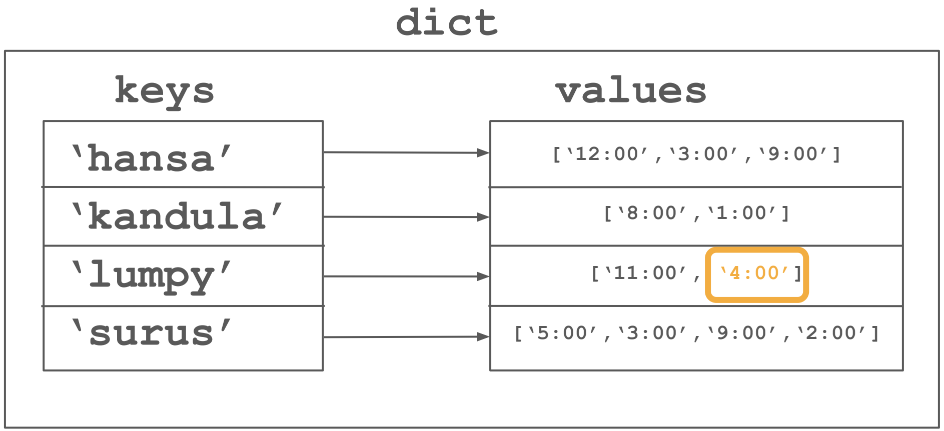
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Add a feeding time (‘4:00’) to ‘lumpy’!*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
```



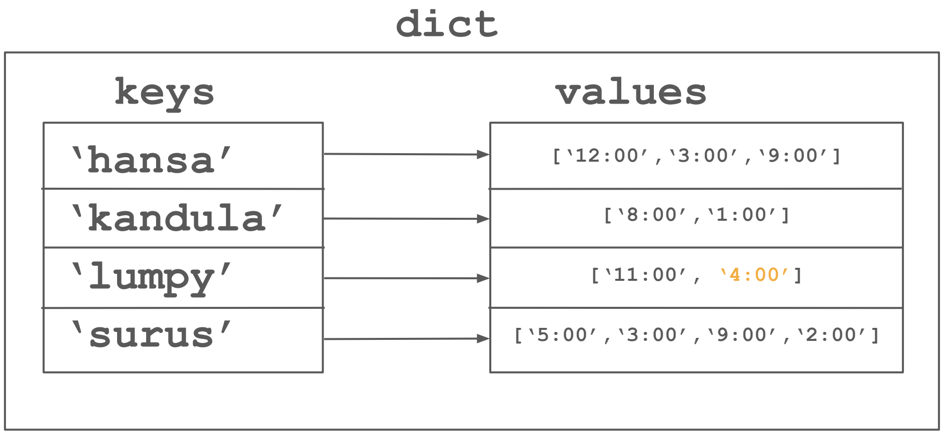
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’,‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
```



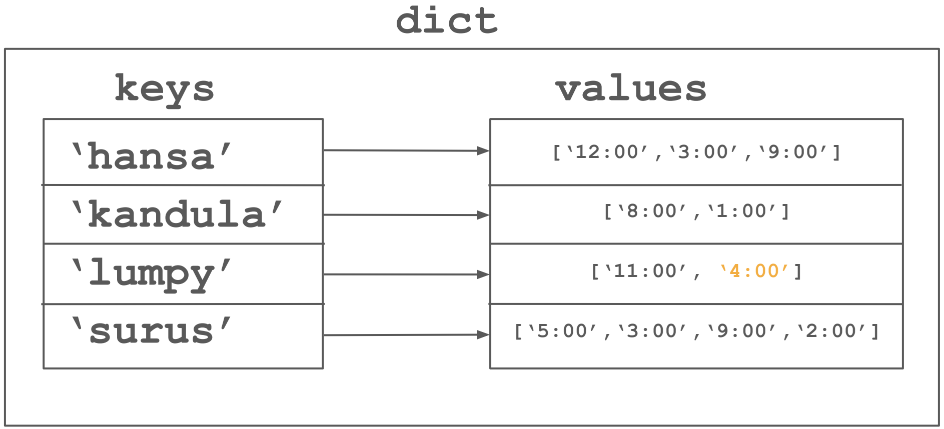
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Get Elem

```
>>> d['hansa']
['12:00','3:00','9:00']
>>> d['lumpy'].append('4:00')
>>> k_times = d['kandula']
['8:00','1:00']
```



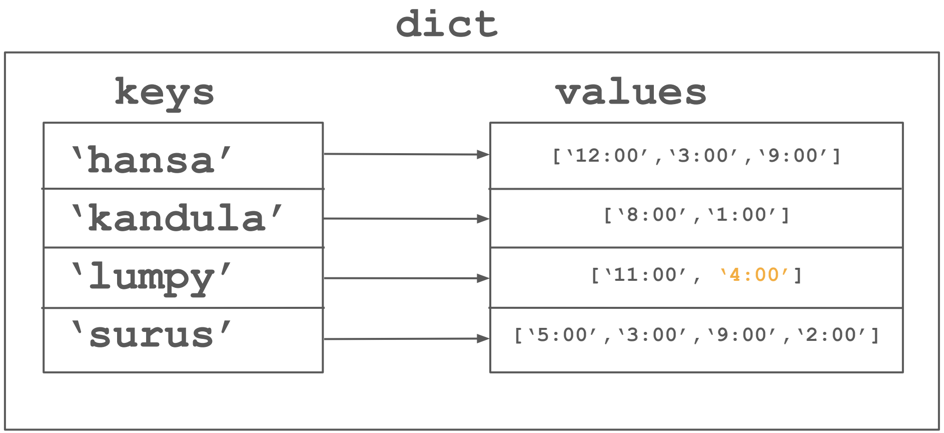
dict

| keys | values |
|---|---|
| 'hansa' | ['12:00','3:00','9:00'] |
| 'kandula' | ['8:00','1:00'] |
| 'lumpy' | ['11:00', '4:00'] |
| 'surus' | ['5:00','3:00','9:00','2:00'] |

*Get the first feeding time for 'kandula'*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
[‘8:00’,‘1:00’]
>>> k_times[0]
```



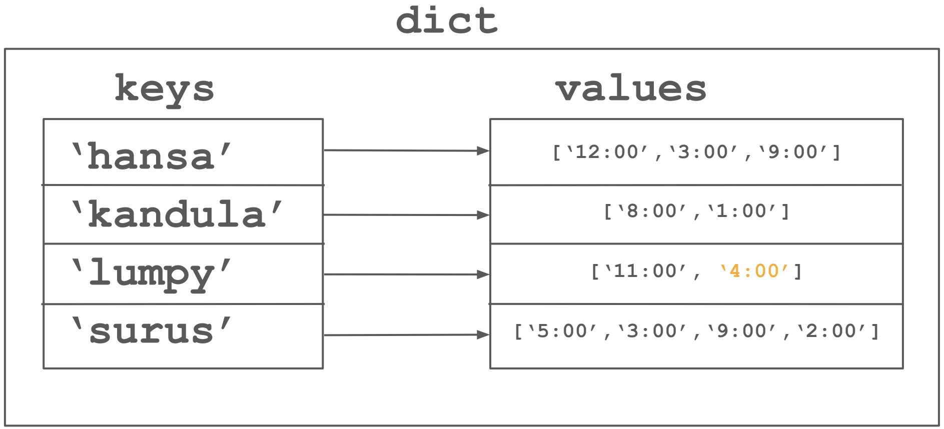
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
[‘8:00’,‘1:00’]
>>> k_times[0]
‘8:00’
```



dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

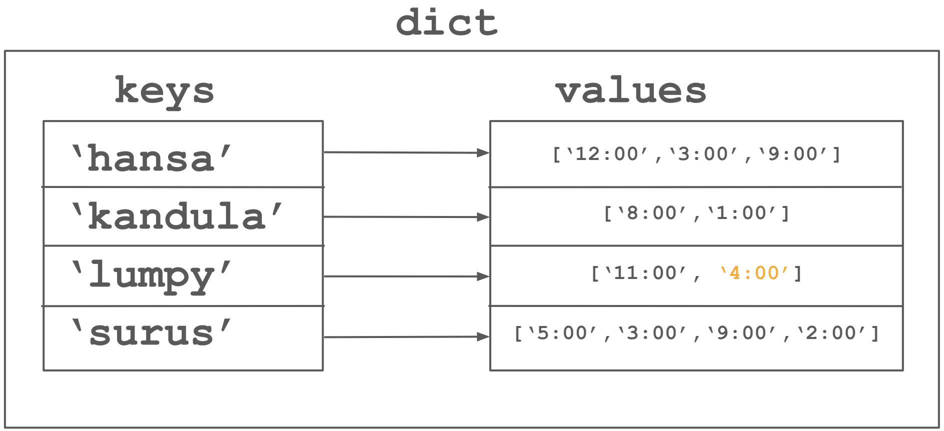
*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
[‘8:00’,‘1:00’]
>>> k_times[0]
‘8:00’
>>> d[‘kandula’][0]
```

More concisely,

Get the first feeding time for ‘kandula’

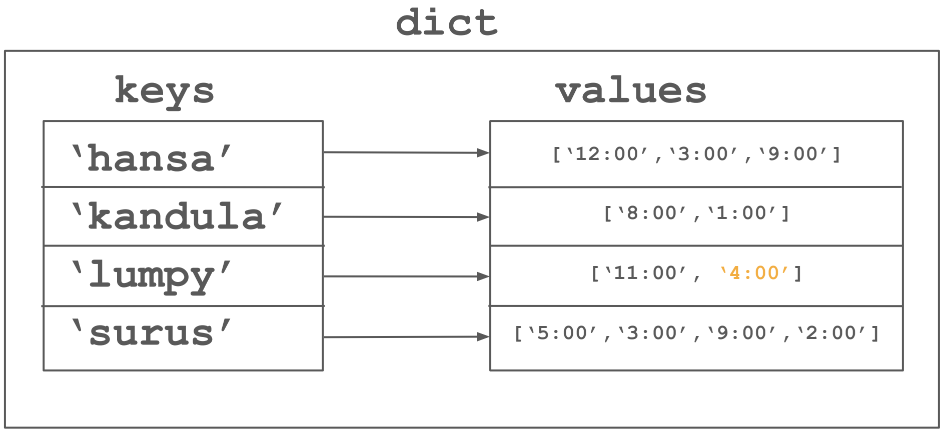


dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
[‘8:00’,‘1:00’]
>>> k_times[0]
‘8:00’
>>> d[‘kandula’][0]
‘8:00’
```



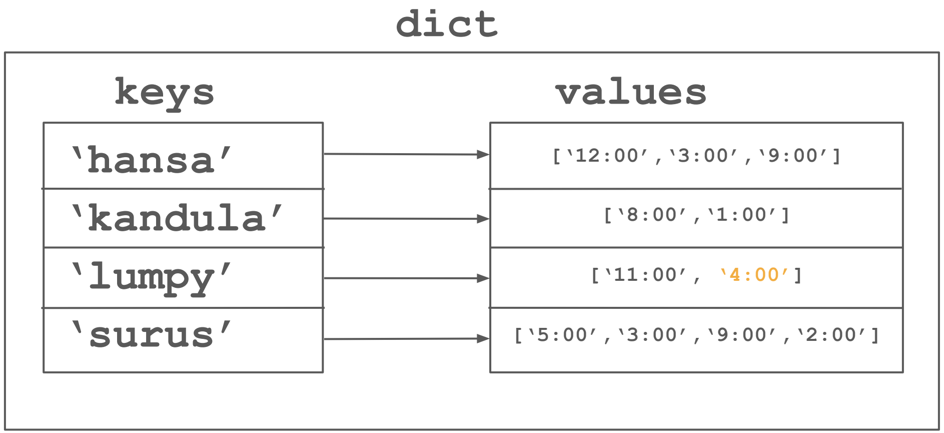
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Get Elem

```
>>> d[‘hansa’]
[‘12:00’,‘3:00’,‘9:00’]
>>> d[‘lumpy’].append(‘4:00’)
>>> k_times = d[‘kandula’]
[‘8:00’,‘1:00’]
>>> k_times[0]
‘8:00’
>>> d[‘kandula’][0]
‘8:00’
```

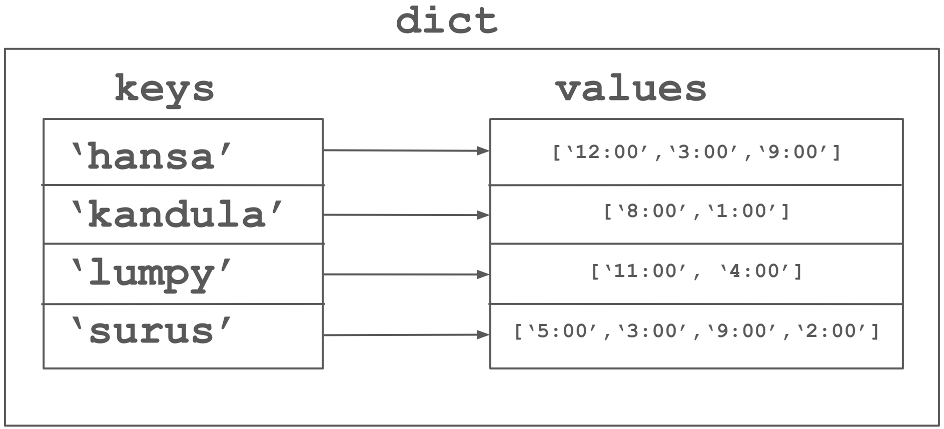


dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Get the first feeding time for ‘kandula’*

# Using a Dictionary Containing a List - Set List



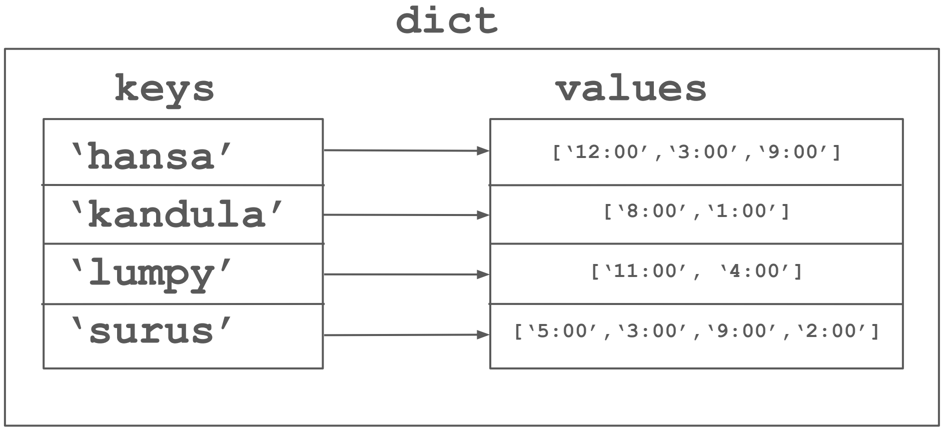
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

Reset ‘surus’ feeding list to [‘7:00’]

# Using a Dictionary Containing a List - Set List

```
>>> d[‘surus’] = [‘7:00’]
```



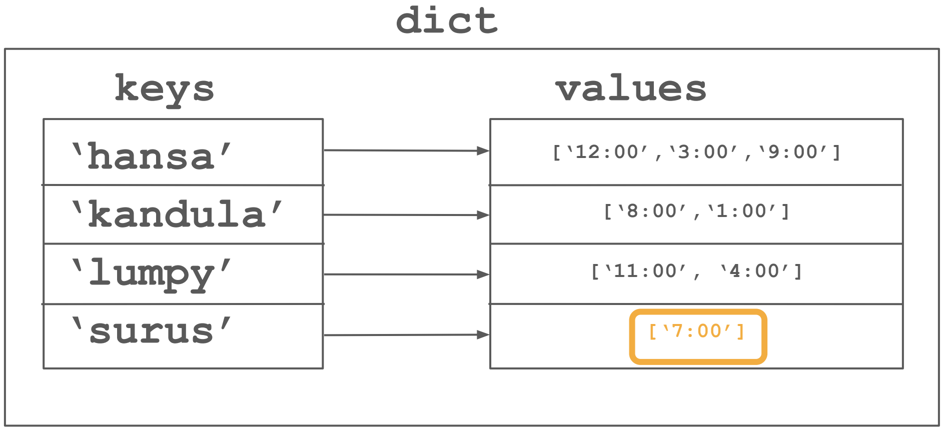
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

*Reset ‘surus’ feeding list to [‘7:00’]*

# Using a Dictionary Containing a List - Set List

```
>>> d[‘surus’] = [‘7:00’]
```



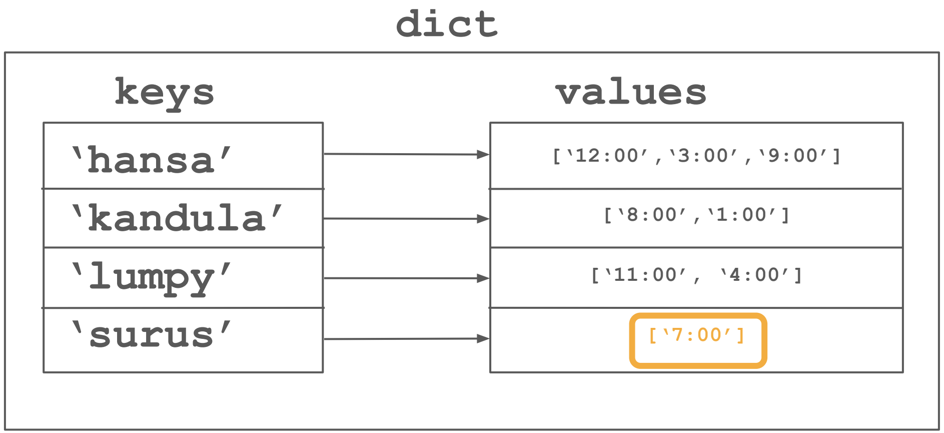
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘7:00’] |

*Reset ‘surus’ feeding list to [‘7:00’]*

# Using a Dictionary Containing a List - Set Element

```
>>> d[‘surus’] = [‘7:00’]
```



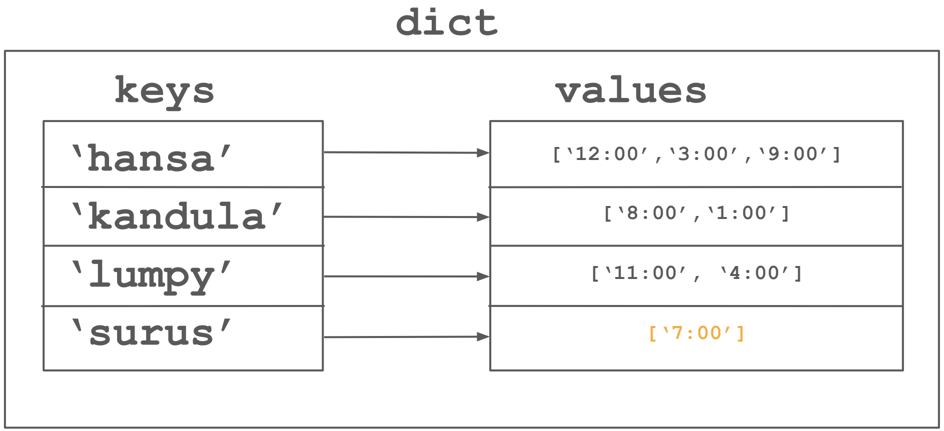
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘7:00’] |

*Set second element in ‘lumpy’ to ‘2:00’*

# Using a Dictionary Containing a List - Set Element

```
>>> d[‘surus’] = [‘7:00’]

>>> lump_list = d[‘lumpy’]
```



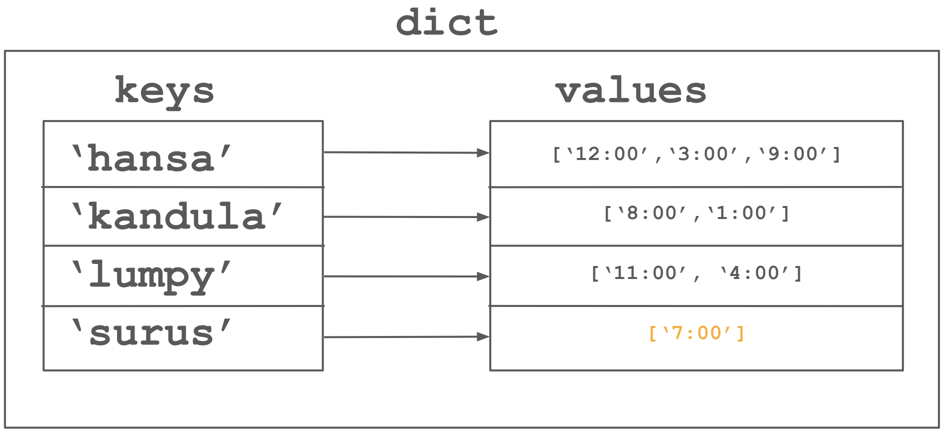
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘7:00’] |

*Set second element in ‘lumpy’ to ‘2:00’*

# Using a Dictionary Containing a List - Set Element

```
>>> d[‘surus’] = [‘7:00’]
>>> lump_list = d[‘lumpy’]
>>> lump_list[1] = ‘2:00’
```



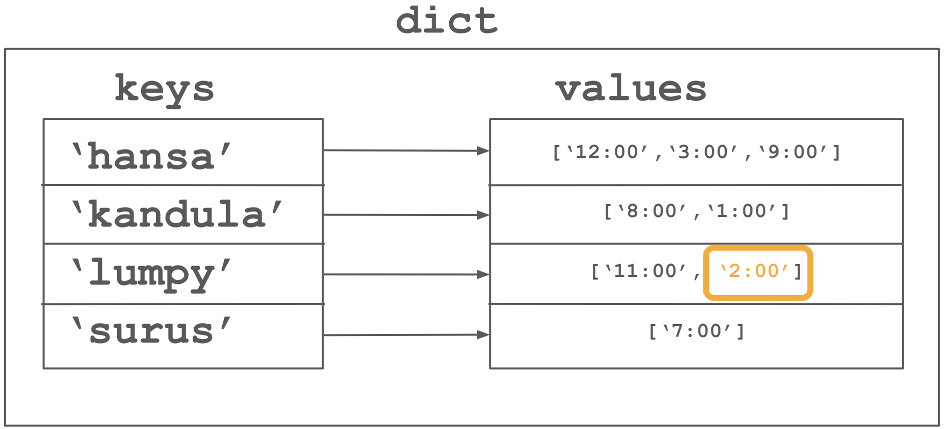
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘4:00’] |
| ‘surus’ | [‘7:00’] |

Set second element in ‘lumpy’ to ‘2:00’

# Using a Dictionary Containing a List - Set Element

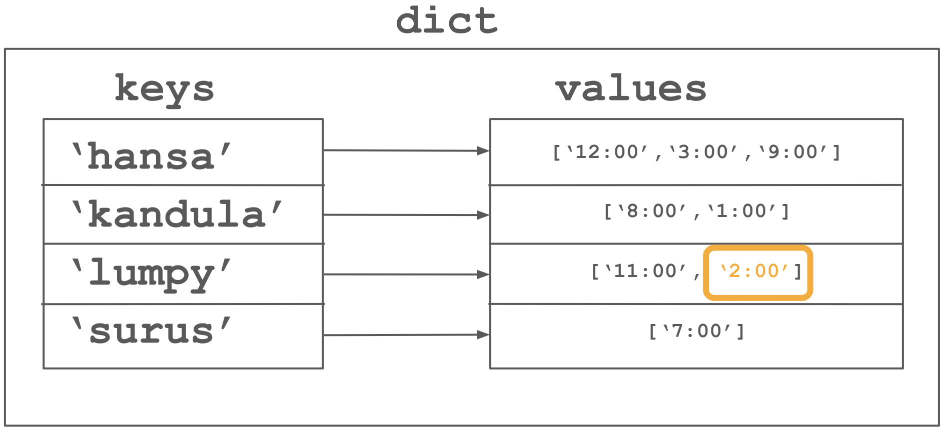
```
>>> d['surus'] = ['7:00']
>>> lump_list = d['lumpy']
>>> lump_list[1] = '2:00'
```



*Set second element in 'lumpy' to '2:00'*

# Using a Dictionary Containing a List - Set Element

```
>>> d[‘surus’] = [‘7:00’]

>>> lump_list = d[‘lumpy’]

>>> lump_list[1] = ‘2:00’

# This is the same thing as:
```



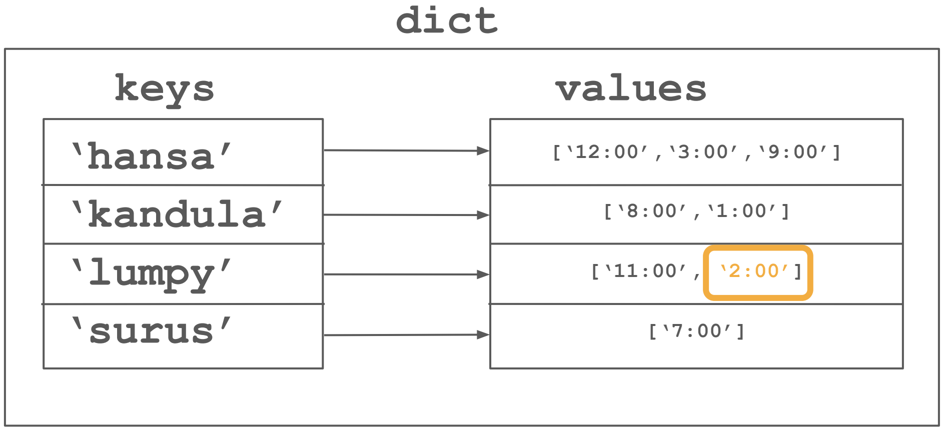
dict

| keys | values |
|---|---|
| ‘hansa’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘11:00’, ‘2:00’] |
| ‘surus’ | [‘7:00’] |

Set second element in ‘lumpy’ to ‘2:00’

# Using a Dictionary Containing a List - Set Element

```
>>> d['surus'] = ['7:00']
>>> lump_list = d['lumpy']
>>> lump_list[1] = '2:00'
# This is the same thing as:
>>> d['lumpy'][1] = '2:00'
```



Set second element in 'lumpy' to '2:00'

# Think/Pair/Share:

How can we modify our file-reading function to populate the animal – feeding times dictionary?

# General Note on Mutability

- Lists and dicts are both mutable data types

# General Note on Mutability

- Lists and dicts are both mutable data types
    - We can append or set, and these will modify the original object

# General Note on Mutability

- Lists and dicts are both mutable data types
    - We can append or set, and these will modify the original object
    - If we pass a list or a dict into a function and modify it, our changes will persist.

# General Note on Mutability

- Lists and dicts are both mutable data types
  - We can append or set, and these will modify the original object
  - If we pass a list or a dict into a function and modify it, our changes will persist. [DEMO]

# General Note on Mutability

- Lists and dicts are both mutable data types
  - We can append or set, and these will modify the original object
  - If we pass a list or a dict into a function and modify it, our changes will persist.
- Only immutable types can be used as dictionary **keys**

# General Note on Mutability

- Lists and dicts are both mutable data types
  - We can append or set, and these will modify the original object
  - If we pass a list or a dict into a function and modify it, our changes will persist.
- Only immutable types can be used as dictionary **keys**
  - e.g. strings, ints, floats, booleans

# General Note on Mutability

- Lists and dicts are both mutable data types
  - We can append or set, and these will modify the original object
  - If we pass a list or a dict into a function and modify it, our changes will persist.
- Only immutable types can be used as dictionary **keys**
  - e.g. strings, ints, floats, booleans
  - immutable or mutable types can be dictionary **values**

# General Note on Mutability

- Lists and dicts are both mutable data types
  - We can append or set, and these will modify the original object
  - If we pass a list or a dict into a function and modify it, our changes will persist.
- Only immutable types can be used as dictionary **keys**
  - e.g. strings, ints, floats, booleans
  - immutable or mutable types can be dictionary **values**
    - e.g. strings, ints, floats, booleans, lists, dictionaries

# Think/Pair/Share:

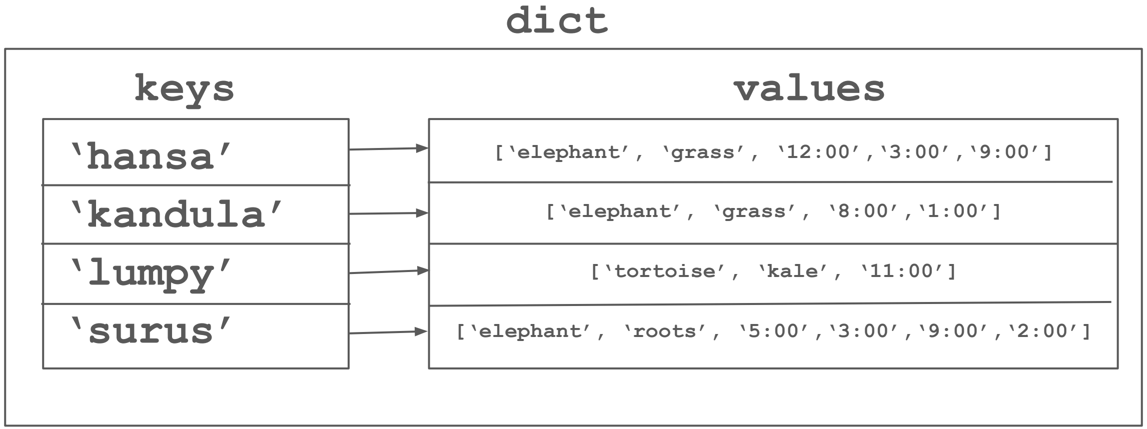
How could we store an animal’s type, diet, and feeding times in a data structure?

# Attempt #1: Animal –  Info List Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶ **list**

# Attempt #1: Animal – Info List Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶ **list**

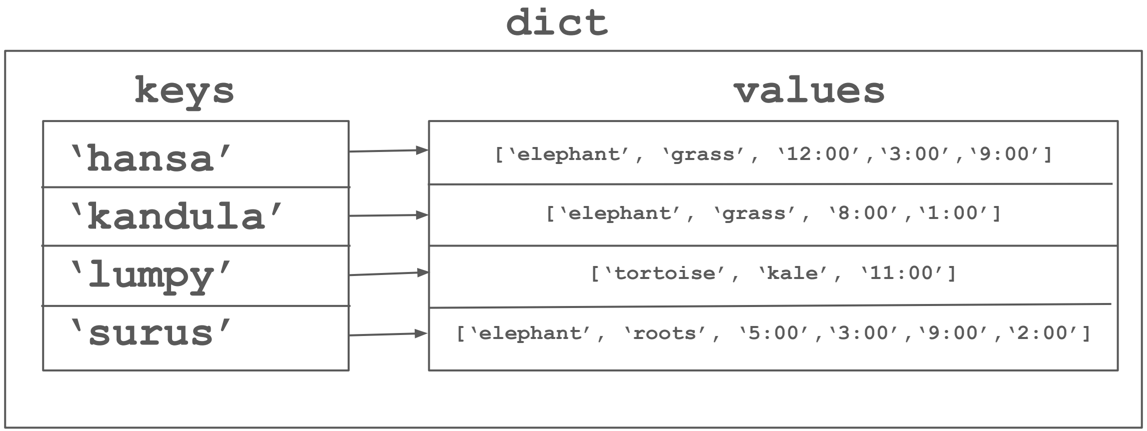


dict

| keys | values |
|---|---|
| ‘hansa’ | [‘elephant’, ‘grass’, ‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘elephant’, ‘grass’, ‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘tortoise’, ‘kale’, ‘11:00’] |
| ‘surus’ | [‘elephant’, ‘roots’, ‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

# Attempt #1: Animal –  Info List Dictionary

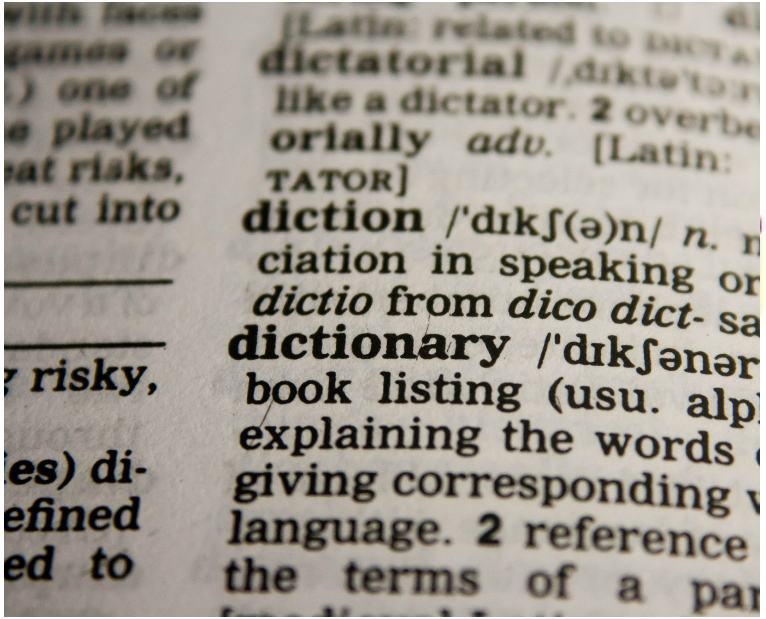
- animal name ⟶ **animal type, diet, feeding times**
- string ⟶  **list**



| keys | values |
|---|---|
| ‘hansa’ | [‘elephant’, ‘grass’, ‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | [‘elephant’, ‘grass’, ‘8:00’,‘1:00’] |
| ‘lumpy’ | [‘tortoise’, ‘kale’, ‘11:00’] |
| ‘surus’ | [‘elephant’, ‘roots’, ‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

✗ Not super easy to distinguish between the different pieces of data in the list

# Dicts in Dicts!

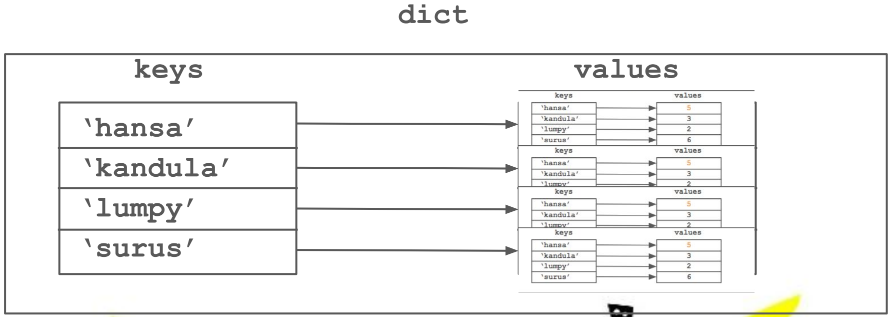


with faces
games or
) one of
e played
at risks,
cut into

risky,

es) di-
efined
ed to

[Latin: related to DICTA
dictatorial /ˌdɪktəˈtɔːr
like a dictator. 2 overbe
orially adv. [Latin:
TATOR]
diction /ˈdɪkʃ(ə)n/ n. m
ciation in speaking or
dictio from dico dict- sa
dictionary /ˈdɪkʃənər
book listing (usu. alp
explaining the words
giving corresponding w
language. 2 reference
the terms of a par



dict

keys | values

'hansa'
'kandula'
'lumpy'
'surus'

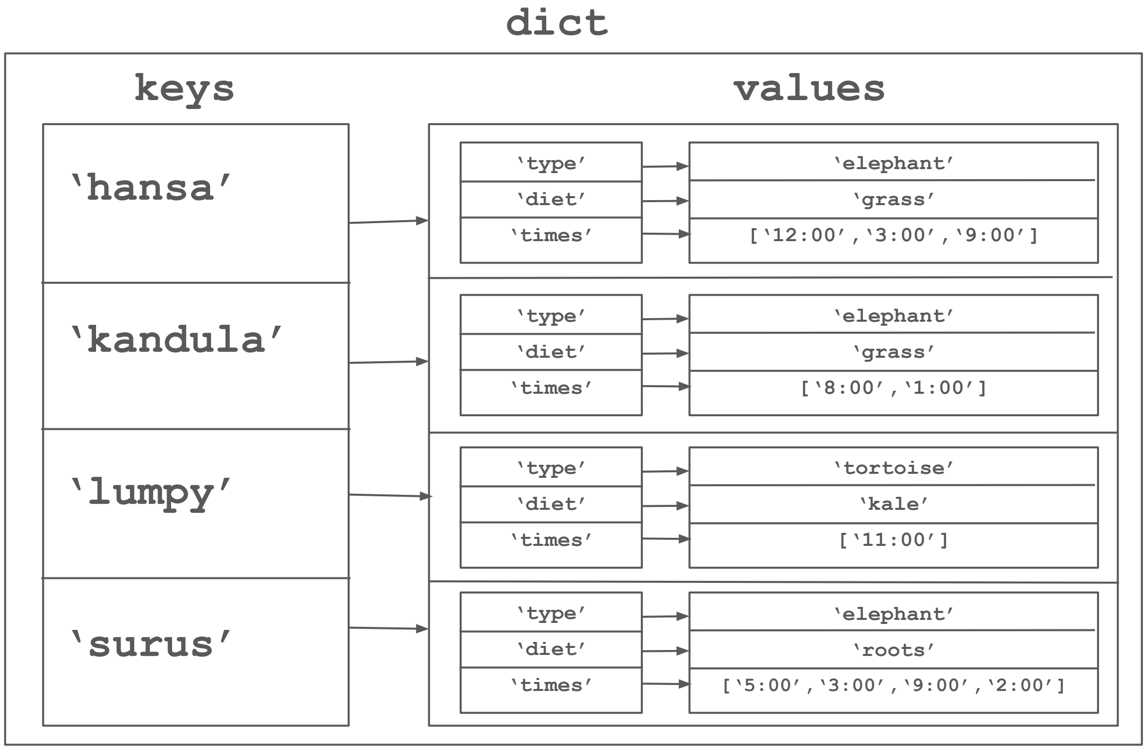
Image from Allie Brosh

# Attempt #2: Animal –  Info Dict Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶  **dict**
- use strings as keys to specify what field the values correspond to

# Attempt #2: Animal – Info Dict Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶ **dict**
- use strings as keys to specify what field the values correspond to



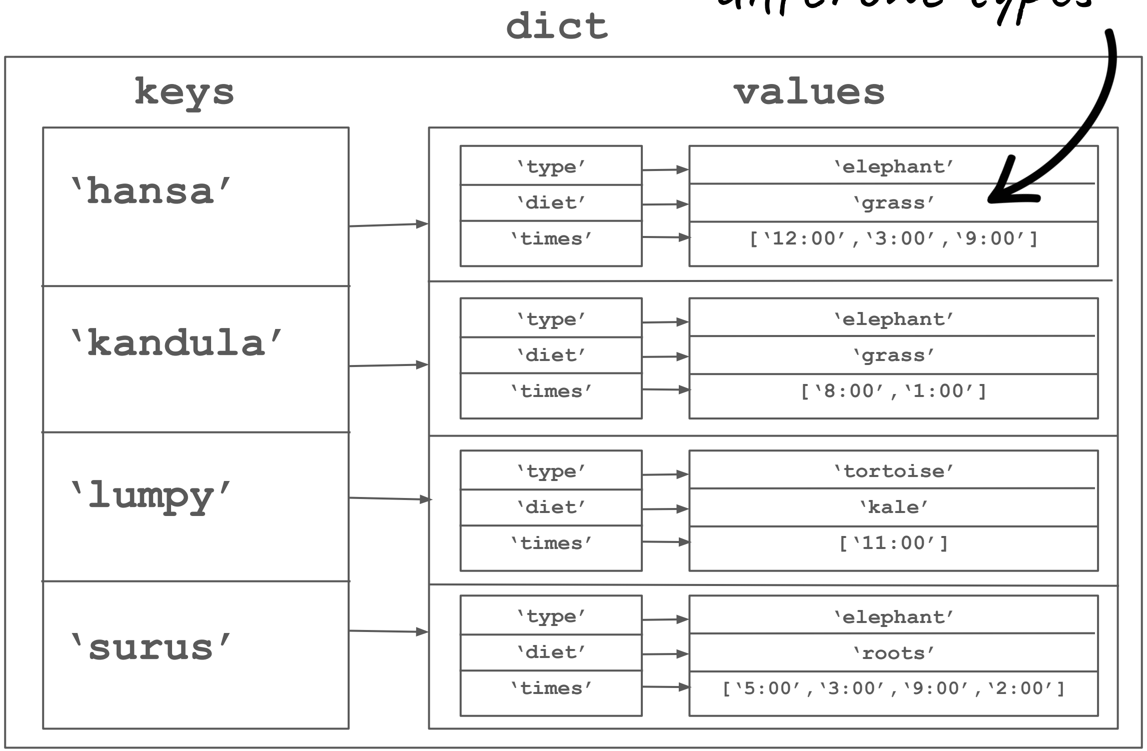
dict

| keys | values | |
|---|---|---|
| ‘hansa’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘grass’ |
| | ‘times’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘grass’ |
| | ‘times’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | ‘type’ | ‘tortoise’ |
| | ‘diet’ | ‘kale’ |
| | ‘times’ | [‘11:00’] |
| ‘surus’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘roots’ |
| | ‘times’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

# Attempt #2: Animal – Info Dict Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶ **dict**
- use strings as keys to specify what field the values correspond to

you can have values of different types



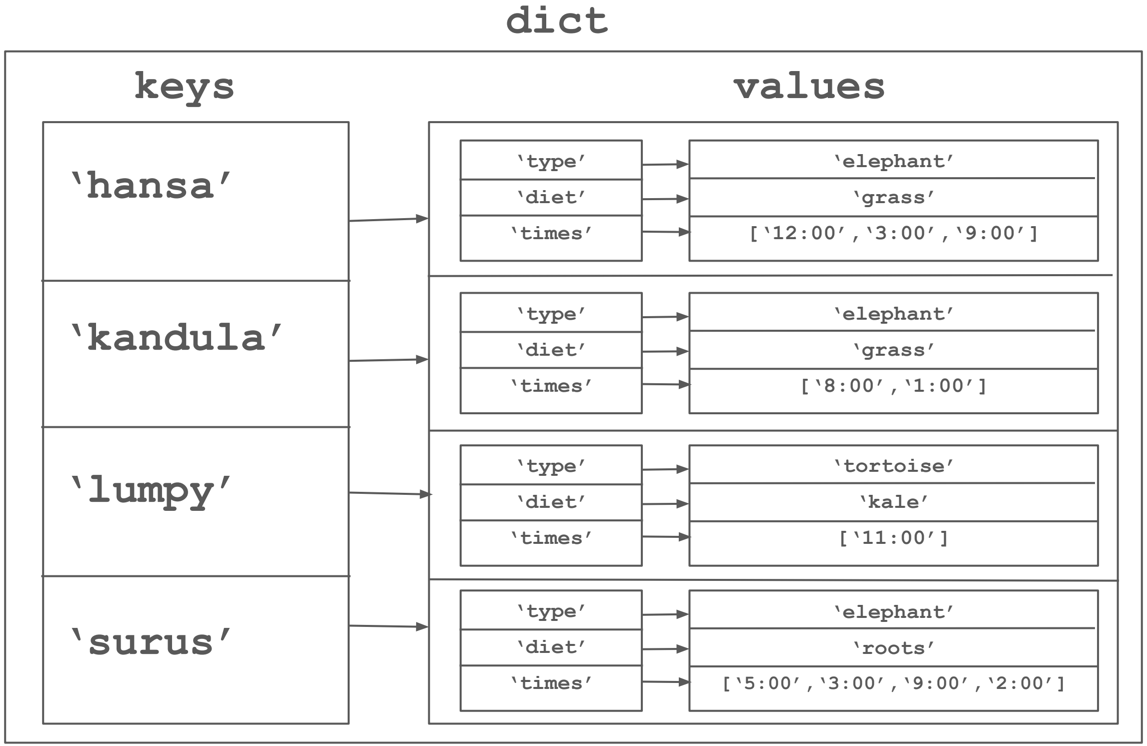
dict

| keys | values | |
|---|---|---|
| ‘hansa’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘grass’ |
| | ‘times’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘grass’ |
| | ‘times’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | ‘type’ | ‘tortoise’ |
| | ‘diet’ | ‘kale’ |
| | ‘times’ | [‘11:00’] |
| ‘surus’ | ‘type’ | ‘elephant’ |
| | ‘diet’ | ‘roots’ |
| | ‘times’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

# Attempt #2: Animal – Info Dict Dictionary

- animal name ⟶ **animal type, diet, feeding times**
- string ⟶ **dict**
- use strings as keys to specify what field the values correspond to

Common pattern



dict

keys

values

'hansa' → 'type' → 'elephant', 'diet' → 'grass', 'times' → ['12:00','3:00','9:00']

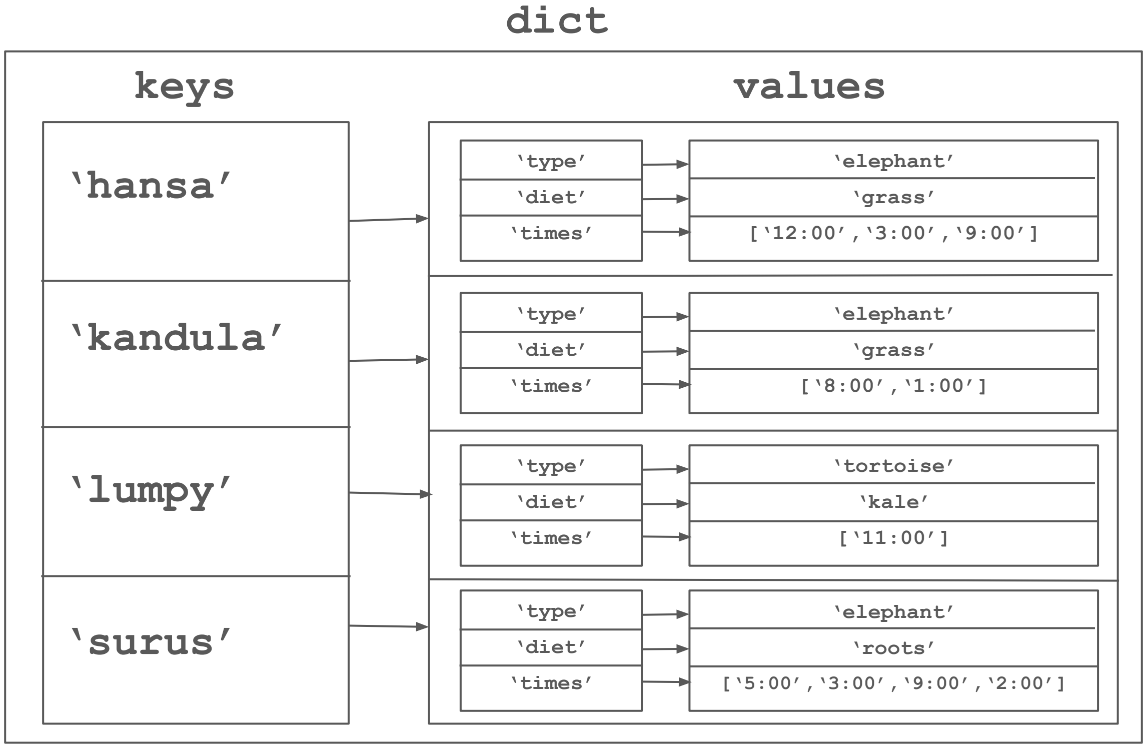
'kandula' → 'type' → 'elephant', 'diet' → 'grass', 'times' → ['8:00','1:00']

'lumpy' → 'type' → 'tortoise', 'diet' → 'kale', 'times' → ['11:00']

'surus' → 'type' → 'elephant', 'diet' → 'roots', 'times' → ['5:00','3:00','9:00','2:00']

# Using a Dictionary Containing a Dict - Get

```
>>> d['hansa']
```



dict

keys

values

'hansa'

'type' → 'elephant'
'diet' → 'grass'
'times' → ['12:00','3:00','9:00']

'kandula'

'type' → 'elephant'
'diet' → 'grass'
'times' → ['8:00','1:00']

'lumpy'

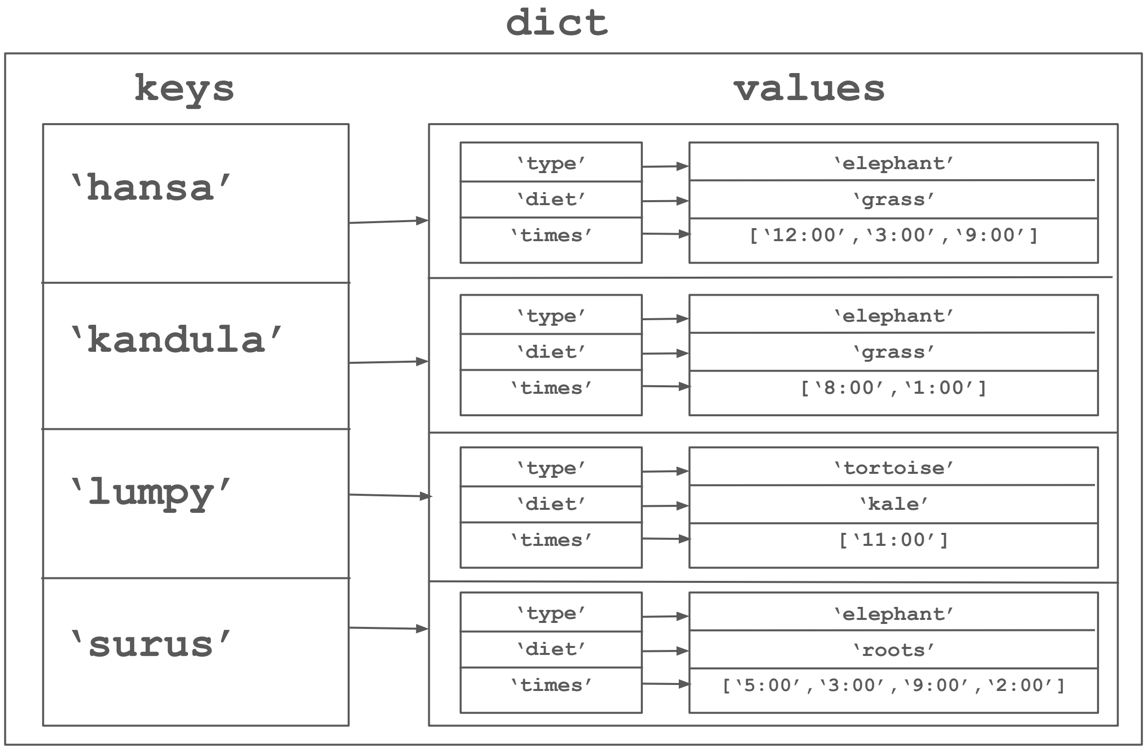
'type' → 'tortoise'
'diet' → 'kale'
'times' → ['11:00']

'surus'

'type' → 'elephant'
'diet' → 'roots'
'times' → ['5:00','3:00','9:00','2:00']

# Using a Dictionary Containing a Dict - Get

```
>>> d[‘hansa’]
{‘type’: ‘elephant’,
‘diet’: ‘grass’,
‘times’: [‘12:00’,‘3:00’,‘9:00’]}
```



dict

keys

values

‘hansa’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘8:00’,‘1:00’]

‘lumpy’

‘type’ → ‘tortoise’

‘diet’ → ‘kale’

‘times’ → [‘11:00’]

‘surus’

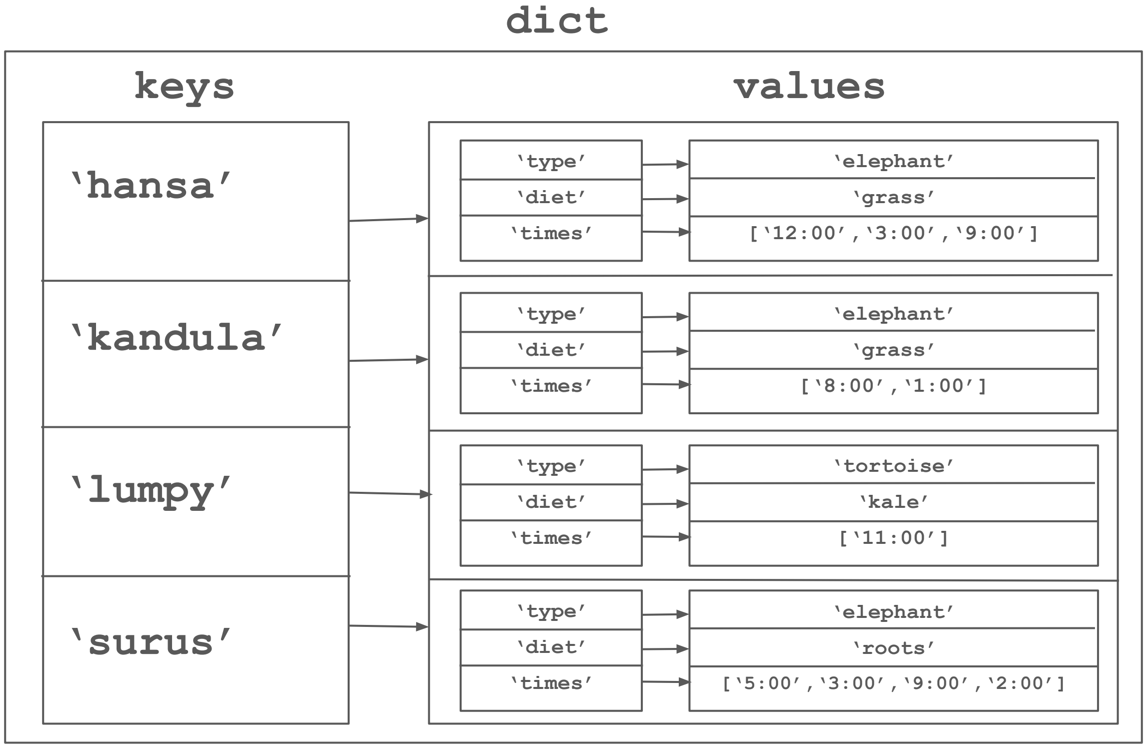
‘type’ → ‘elephant’

‘diet’ → ‘roots’

‘times’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Get

```
>>> d[‘hansa’]
{‘type’: ‘elephant’,
‘diet’: ‘grass’,
‘times’: [‘12:00’,‘3:00’,‘9:00’]}
>>> d[‘hansa’][‘type’]
```



dict

keys

values

‘hansa’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘8:00’,‘1:00’]

‘lumpy’

‘type’ ‘tortoise’

‘diet’ ‘kale’

‘times’ [‘11:00’]

‘surus’

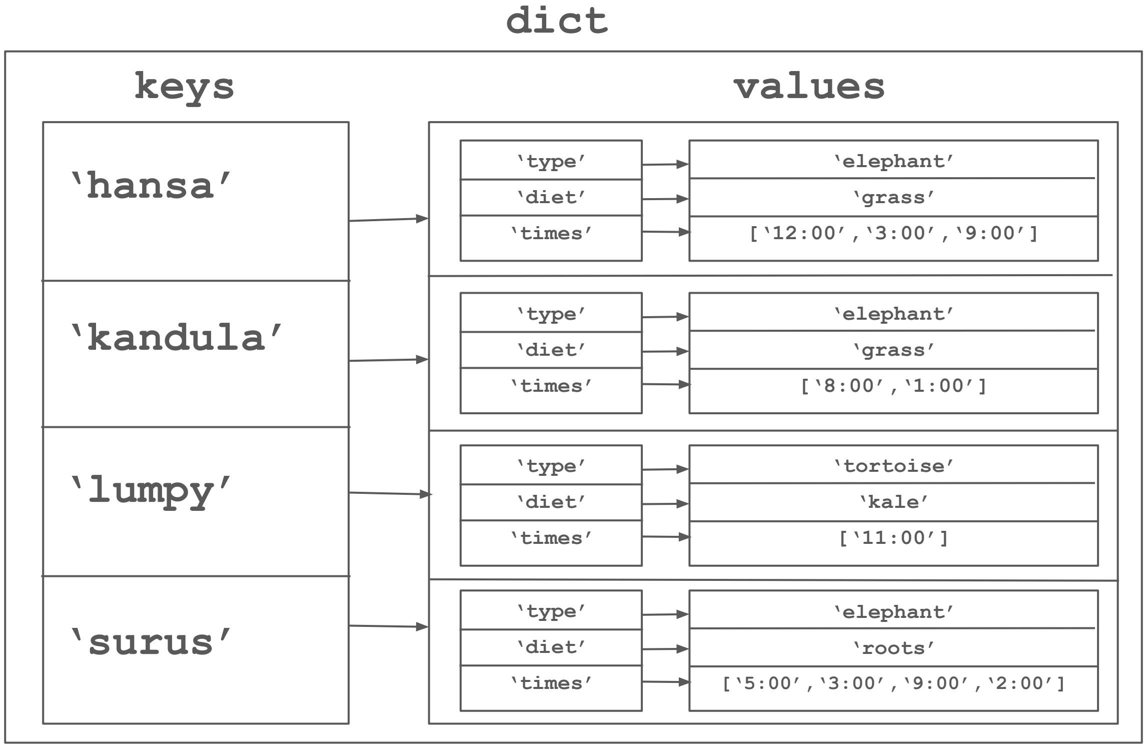
‘type’ ‘elephant’

‘diet’ ‘roots’

‘times’ [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Get

```
>>> d[‘hansa’]
{‘type’: ‘elephant’,
‘diet’: ‘grass’,
‘times’: [‘12:00’,‘3:00’,‘9:00’]}
>>> d[‘hansa’][‘type’]
‘elephant’
```



dict

keys

values

‘hansa’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘8:00’,‘1:00’]

‘lumpy’

‘type’ → ‘tortoise’

‘diet’ → ‘kale’

‘times’ → [‘11:00’]

‘surus’

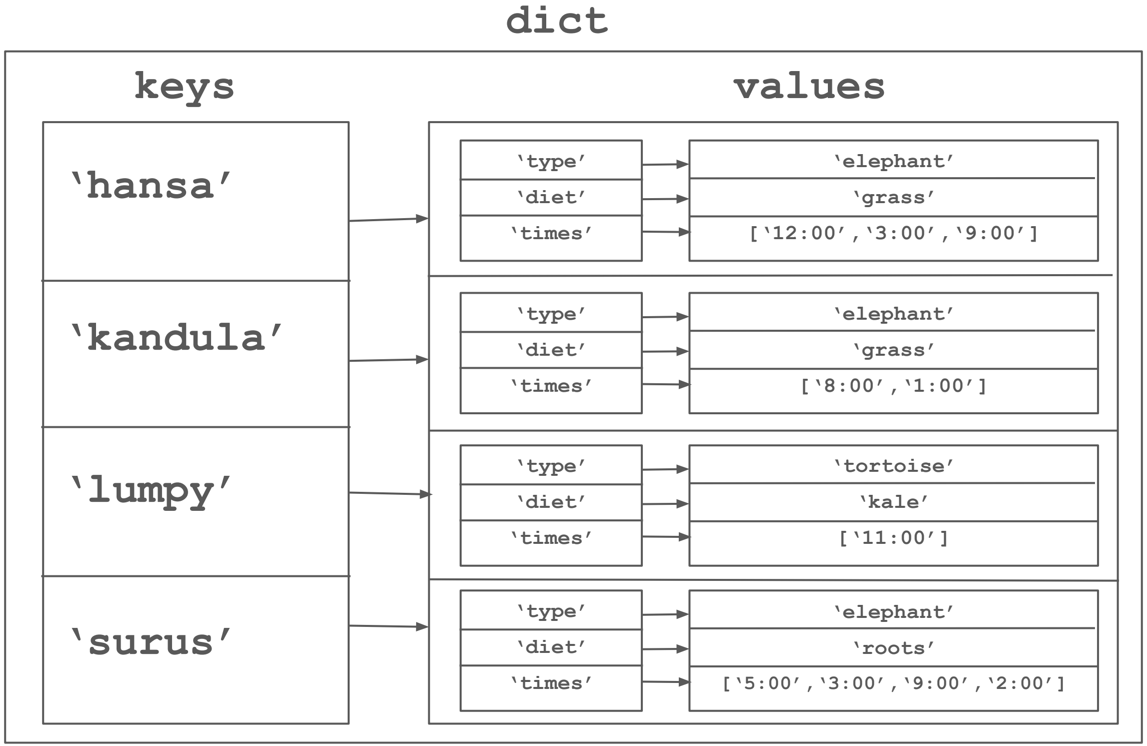
‘type’ → ‘elephant’

‘diet’ → ‘roots’

‘times’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Get

```
>>> d[‘hansa’]
{‘type’: ‘elephant’,
‘diet’: ‘grass’,
‘times’: [‘12:00’,‘3:00’,‘9:00’]}
>>> d[‘hansa’][‘type’]
‘elephant’
>>> d[‘hansa’][‘times’]
```



dict

keys

values

‘hansa’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘8:00’,‘1:00’]

‘lumpy’

‘type’ ‘tortoise’

‘diet’ ‘kale’

‘times’ [‘11:00’]

‘surus’

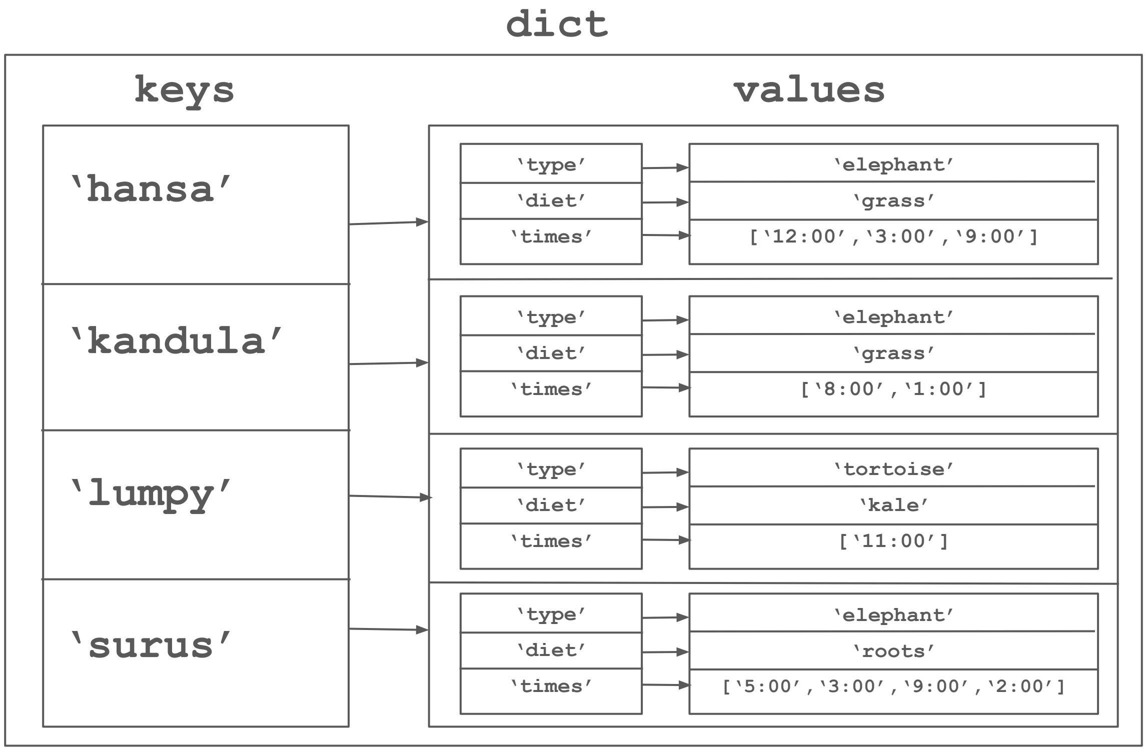
‘type’ ‘elephant’

‘diet’ ‘roots’

‘times’ [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Get

```
>>> d[‘hansa’]
{‘type’: ‘elephant’,
‘diet’: ‘grass’,
‘times’: [‘12:00’,‘3:00’,‘9:00’]}
>>> d[‘hansa’][‘type’]
‘elephant’
>>> d[‘hansa’][‘times’]
[‘12:00’,‘3:00’,‘9:00’]
```



dict

keys

values

‘hansa’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ → ‘elephant’

‘diet’ → ‘grass’

‘times’ → [‘8:00’,‘1:00’]

‘lumpy’

‘type’ → ‘tortoise’

‘diet’ → ‘kale’

‘times’ → [‘11:00’]

‘surus’

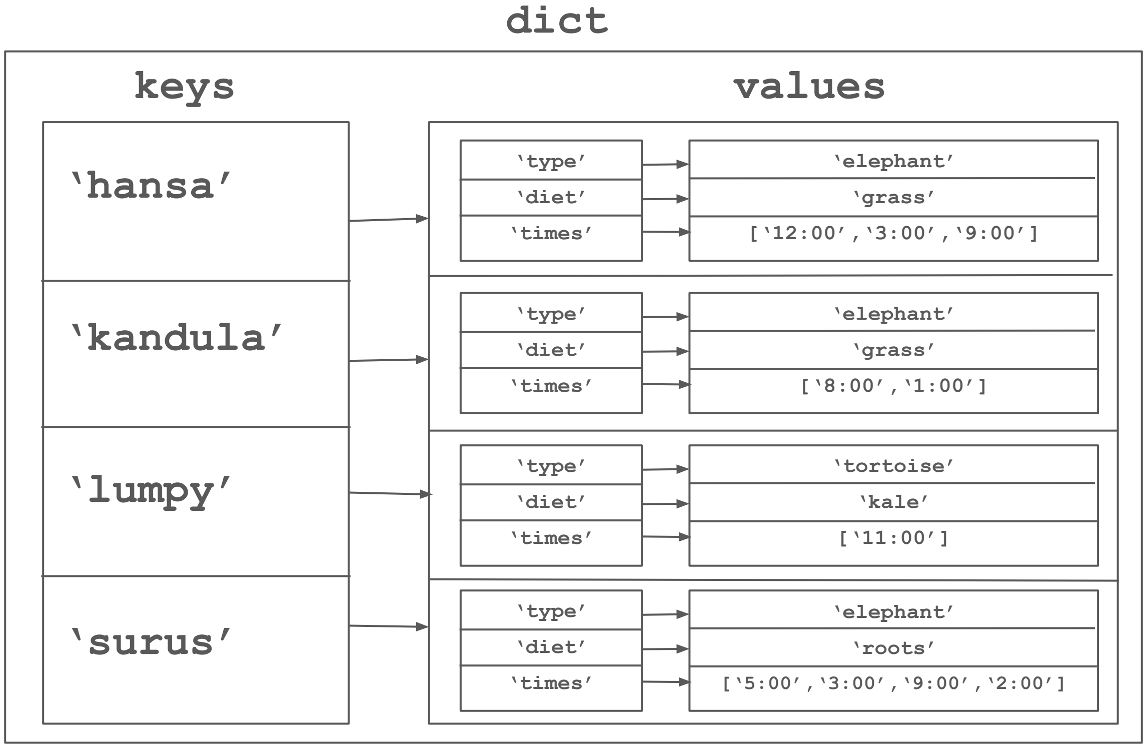
‘type’ → ‘elephant’

‘diet’ → ‘roots’

‘times’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Set

```
# for animal 'sky'

>>> new_dict = {}
```



dict

keys

values

'hansa'

'type' → 'elephant'

'diet' → 'grass'

'times' → ['12:00','3:00','9:00']

'kandula'

'type' → 'elephant'

'diet' → 'grass'

'times' → ['8:00','1:00']

'lumpy'

'type' → 'tortoise'

'diet' → 'kale'

'times' → ['11:00']

'surus'

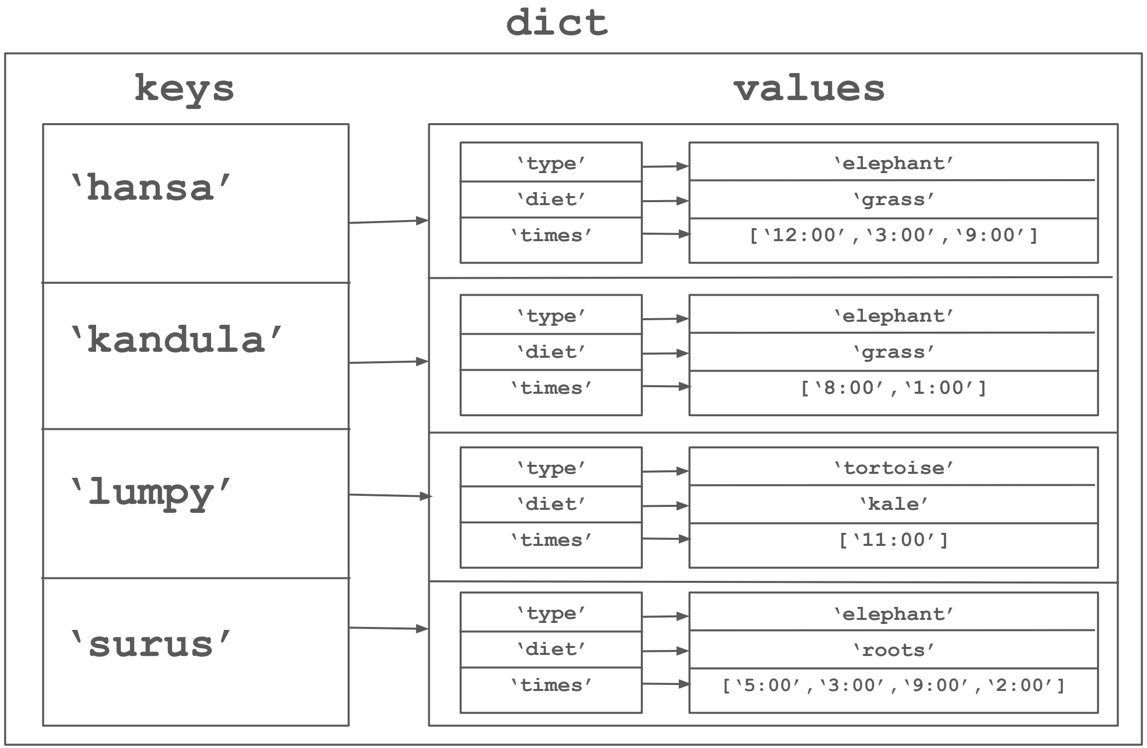
'type' → 'elephant'

'diet' → 'roots'

'times' → ['5:00','3:00','9:00','2:00']

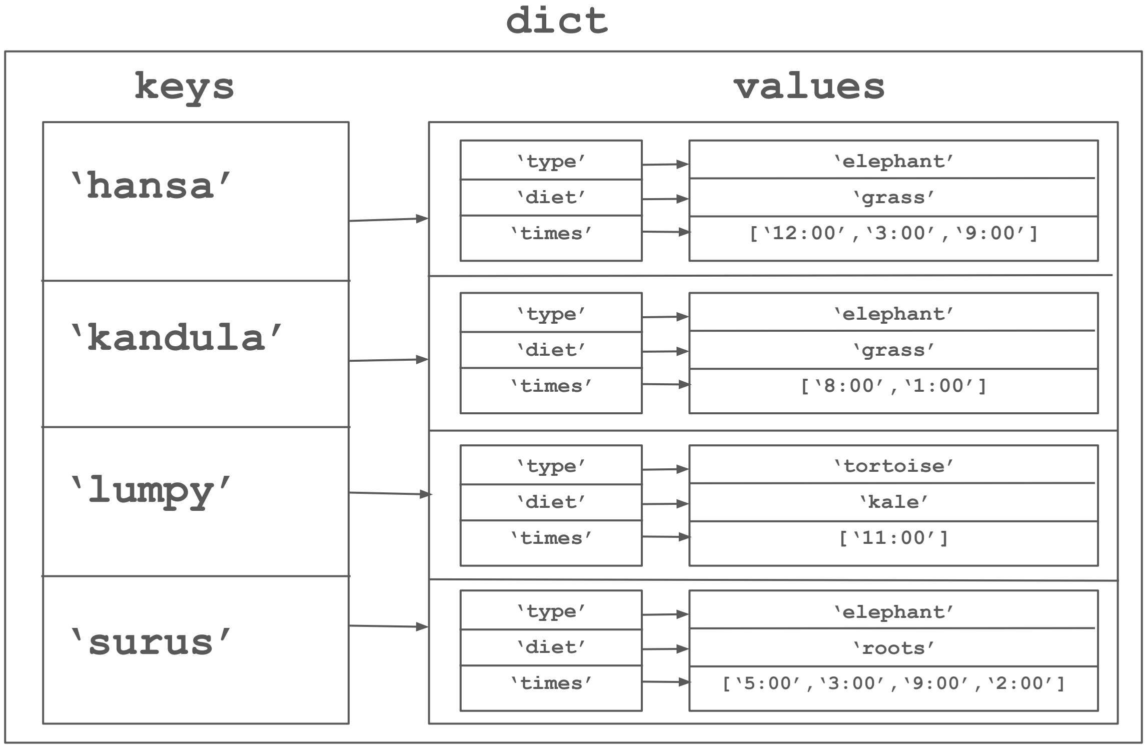
# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’
```



# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’

>>> new_dict[‘diet’] = ‘grass’
```

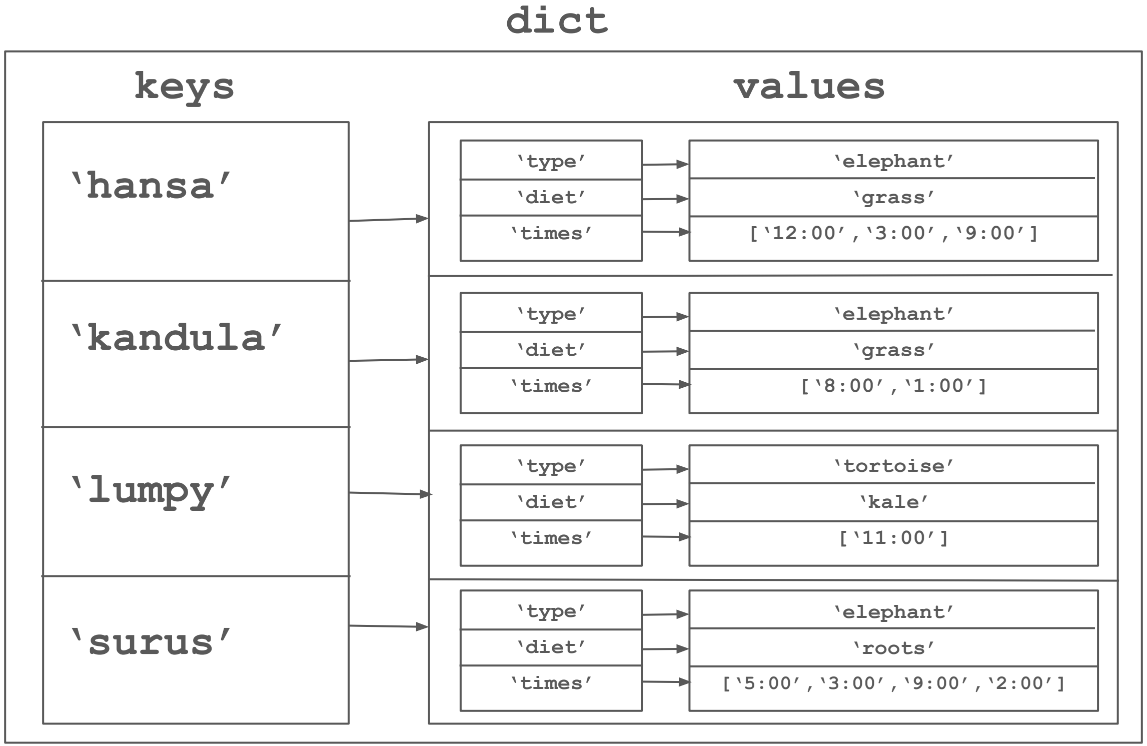


dict

| keys | values |  |
|---|---|---|
| ‘hansa’ | ‘type’ | ‘elephant’ |
|  | ‘diet’ | ‘grass’ |
|  | ‘times’ | [‘12:00’,‘3:00’,‘9:00’] |
| ‘kandula’ | ‘type’ | ‘elephant’ |
|  | ‘diet’ | ‘grass’ |
|  | ‘times’ | [‘8:00’,‘1:00’] |
| ‘lumpy’ | ‘type’ | ‘tortoise’ |
|  | ‘diet’ | ‘kale’ |
|  | ‘times’ | [‘11:00’] |
| ‘surus’ | ‘type’ | ‘elephant’ |
|  | ‘diet’ | ‘roots’ |
|  | ‘times’ | [‘5:00’,‘3:00’,‘9:00’,‘2:00’] |

# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’

>>> new_dict[‘diet’] = ‘grass’

>>> new_dict[‘times’] = [‘4:00’]
```



dict

keys

values

‘hansa’

‘type’ → ‘elephant’
‘diet’ → ‘grass’
‘times’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ → ‘elephant’
‘diet’ → ‘grass’
‘times’ → [‘8:00’,‘1:00’]

‘lumpy’

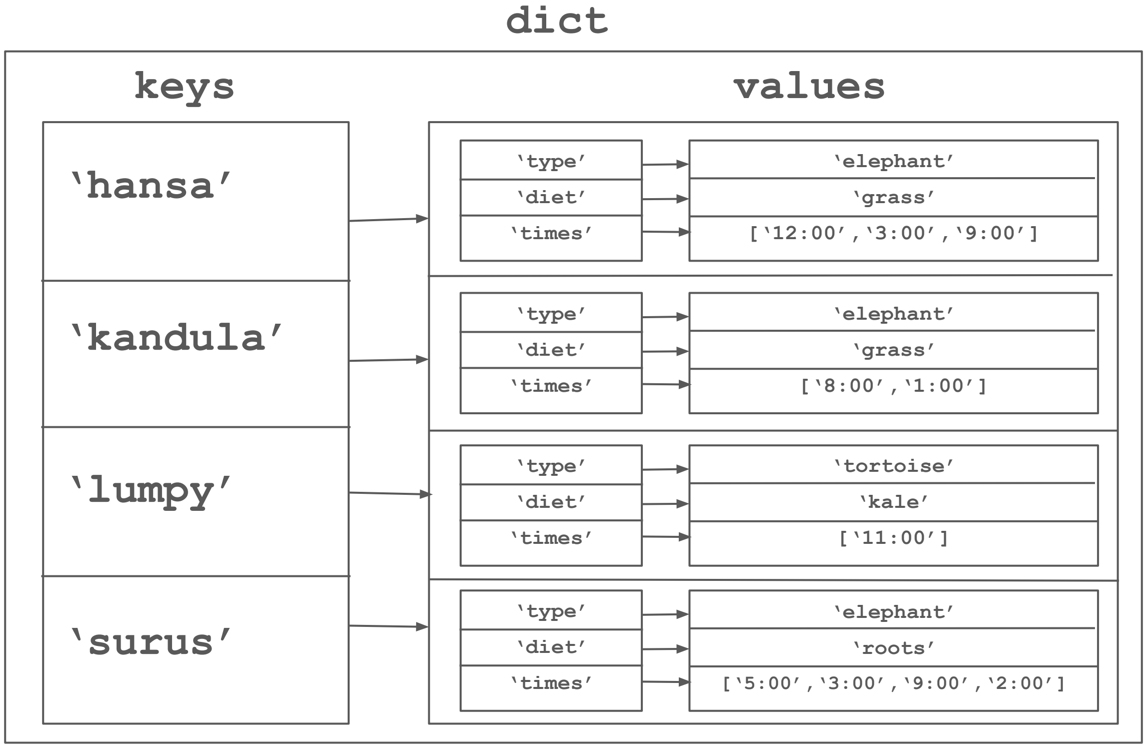
‘type’ → ‘tortoise’
‘diet’ → ‘kale’
‘times’ → [‘11:00’]

‘surus’

‘type’ → ‘elephant’
‘diet’ → ‘roots’
‘times’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’

>>> new_dict[‘diet’] = ‘grass’

>>> new_dict[‘times’] = [‘4:00’]

>>> new_dict
```



dict

keys

values

‘hansa’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ ‘elephant’

‘diet’ ‘grass’

‘times’ [‘8:00’,‘1:00’]

‘lumpy’

‘type’ ‘tortoise’

‘diet’ ‘kale’

‘times’ [‘11:00’]

‘surus’

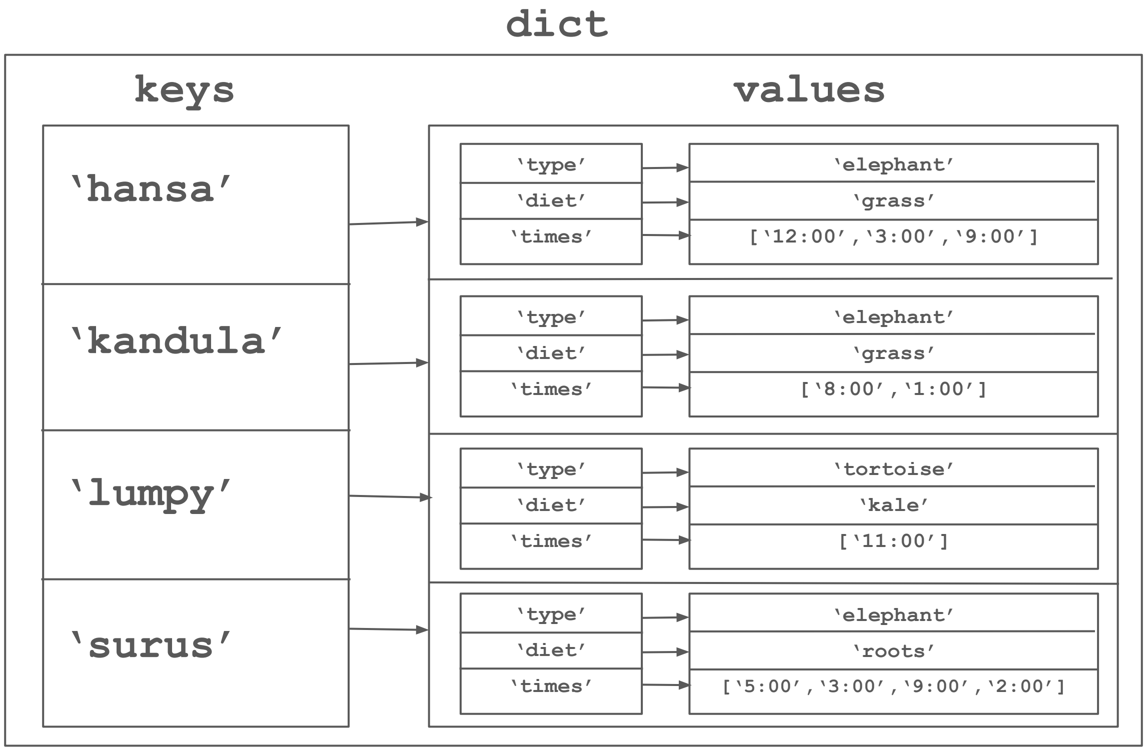
‘type’ ‘elephant’

‘diet’ ‘roots’

‘times’ [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Set

```
# for animal 'sky'

>>> new_dict = {}

>>> new_dict['type'] = 'chicken'

>>> new_dict['diet'] = 'grass'

>>> new_dict['times'] = ['4:00']

>>> new_dict

{'type': 'chicken', 'diet':
'grass', 'times': ['4:00']}
```



dict

keys

values

'hansa'

'type' → 'elephant'
'diet' → 'grass'
'times' → ['12:00','3:00','9:00']

'kandula'

'type' → 'elephant'
'diet' → 'grass'
'times' → ['8:00','1:00']

'lumpy'

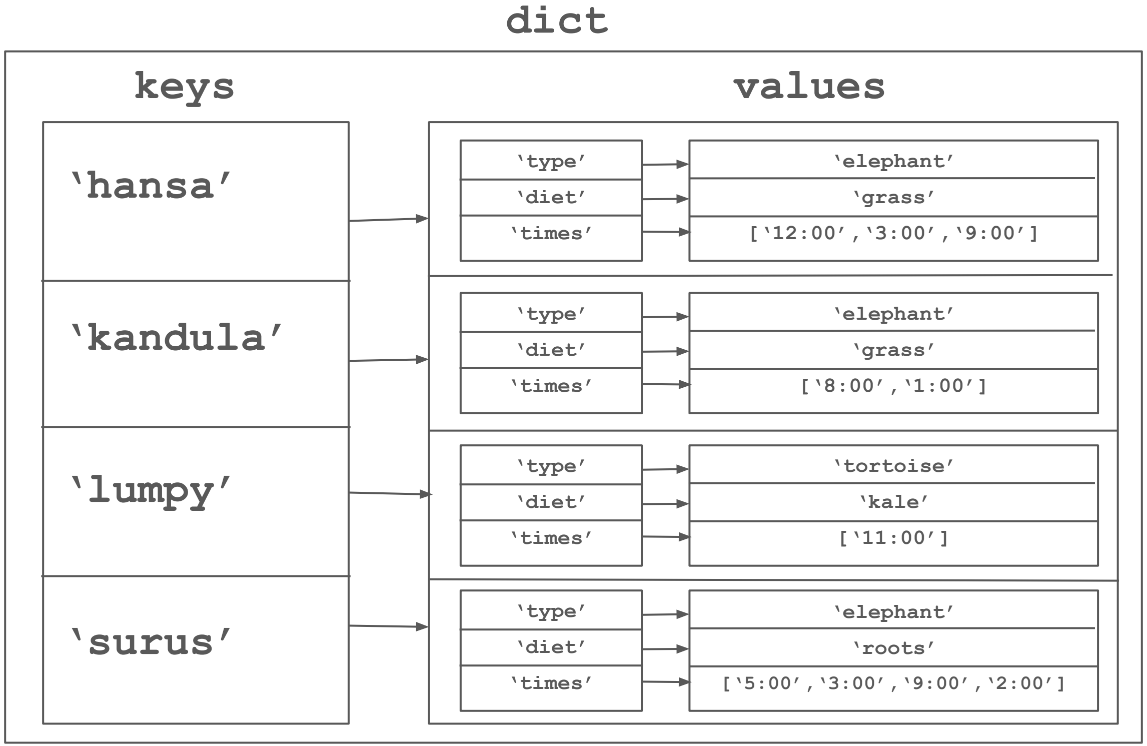
'type' → 'tortoise'
'diet' → 'kale'
'times' → ['11:00']

'surus'

'type' → 'elephant'
'diet' → 'roots'
'times' → ['5:00','3:00','9:00','2:00']

# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’

>>> new_dict[‘diet’] = ‘grass’

>>> new_dict[‘times’] = [‘4:00’]

>>> new_dict

{‘type’: ‘chicken’, ‘diet’:
‘grass’, ‘times’: [‘4:00’]}

>>> d[‘sky’] = new_dict
```



dict

keys

values

‘hansa’

‘type’ → ‘elephant’
‘diet’ → ‘grass’
‘times’ → [‘12:00’,‘3:00’,‘9:00’]

‘kandula’

‘type’ → ‘elephant’
‘diet’ → ‘grass’
‘times’ → [‘8:00’,‘1:00’]

‘lumpy’

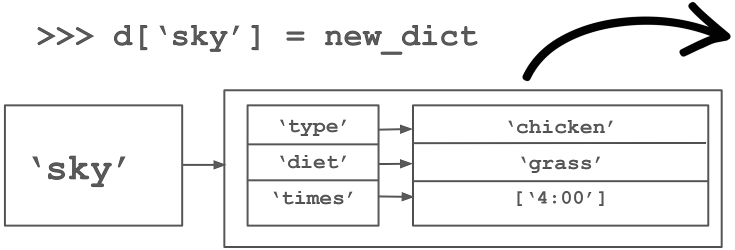
‘type’ → ‘tortoise’
‘diet’ → ‘kale’
‘times’ → [‘11:00’]

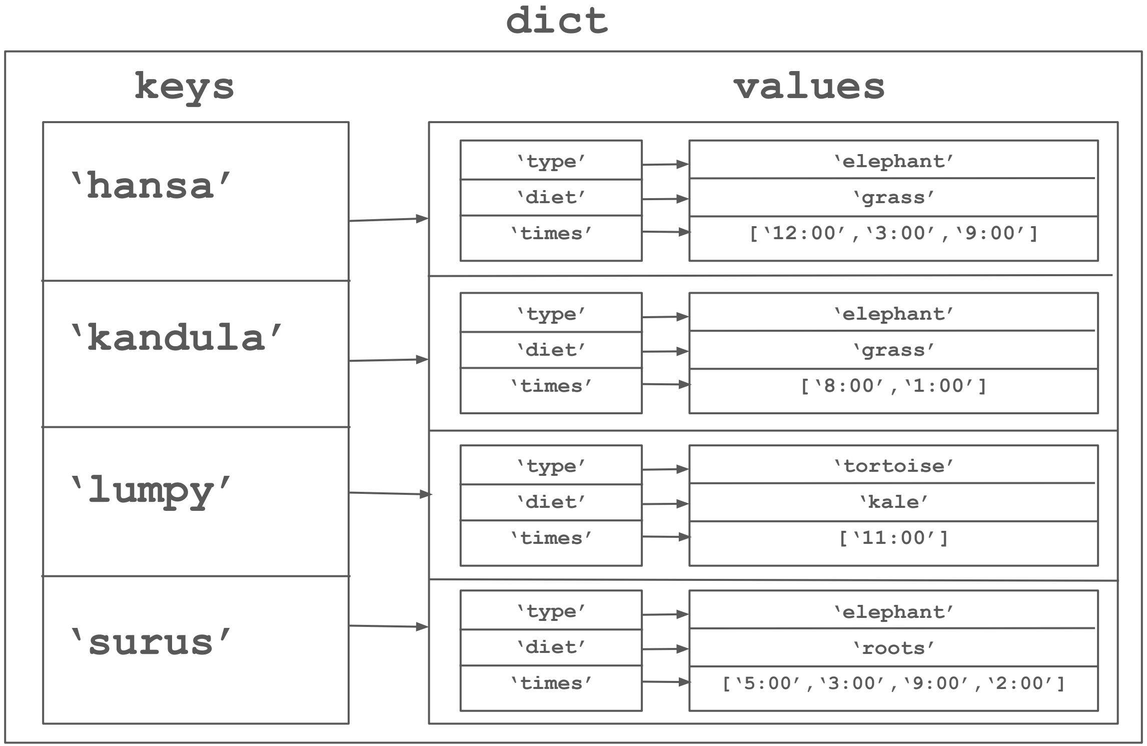
‘surus’

‘type’ → ‘elephant’
‘diet’ → ‘roots’
‘times’ → [‘5:00’,‘3:00’,‘9:00’,‘2:00’]

# Using a Dictionary Containing a Dict - Set

```
# for animal ‘sky’

>>> new_dict = {}

>>> new_dict[‘type’] = ‘chicken’

>>> new_dict[‘diet’] = ‘grass’

>>> new_dict[‘times’] = [‘4:00’]

>>> new_dict

{‘type’: ‘chicken’, ‘diet’:
‘grass’, ‘times’: [‘4:00’]}

>>> d[‘sky’] = new_dict
```





# Nested Data Structures Overview

- We can have lists in lists, dicts in lists, dicts in dicts, and so on...

# Nested Data Structures Overview

- We can have lists in lists, dicts in lists, dicts in dicts, and so on…

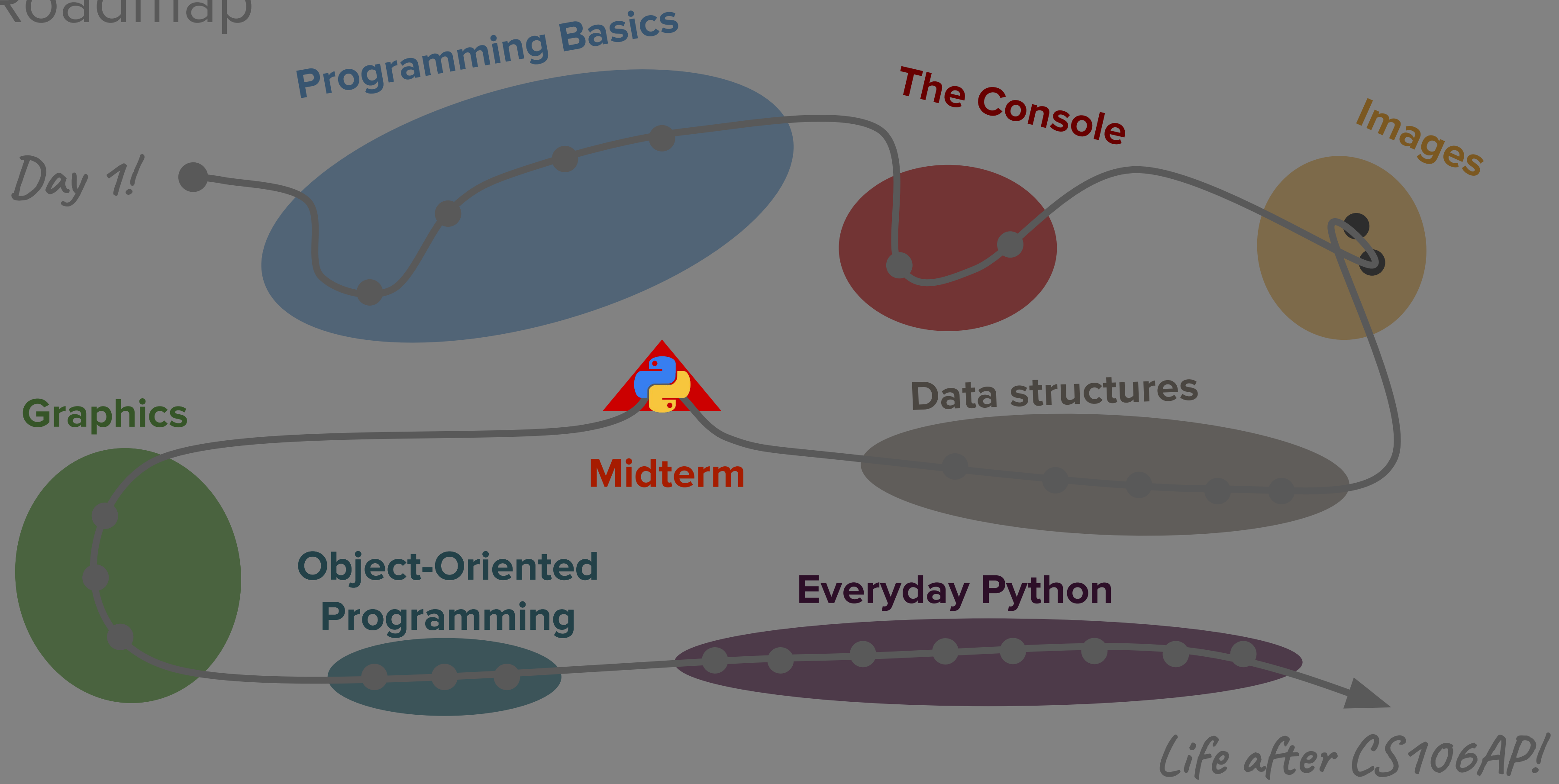
- Lists and dicts are mutable (and can’t be used as **keys**)

# Nested Data Structures Overview

- We can have lists in lists, dicts in lists, dicts in dicts, and so on…
- Lists and dicts are mutable (and can’t be used as **keys**)
- Nesting data structures can help us store even more information in a structured manner!

# What’s next?

# Roadmap


Programming Basics
Day 1!
The Console
Images
Midterm
Data structures
Graphics
Object-Oriented Programming
Everyday Python
Life after CS106AP!