CONDITIONS AND SEQUENCE

PREPARED BY: LUIS MEING

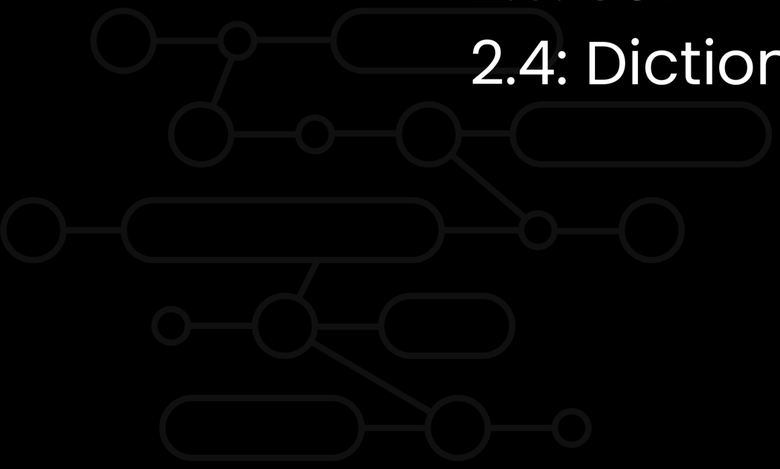
OBJECTIVES:

01 Introduction to Data Structures

1.1: Definition
1.2: Types
1.3: Index

02 Built-in Python Data Structures

2.1: List
2.2: Tuple
2.3: Set
2.4: Dictionary



1.1 Definition

Data Structures

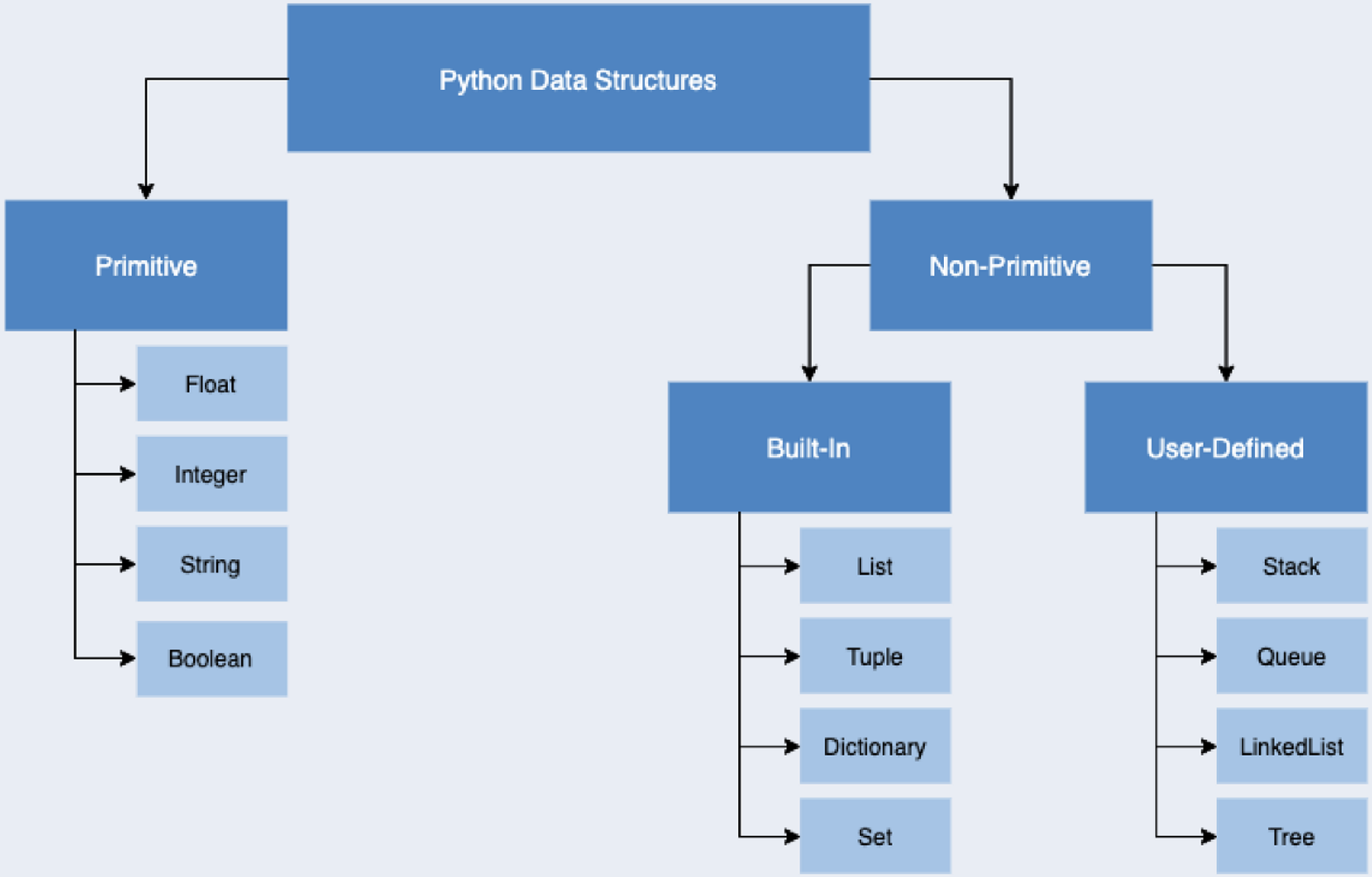
- It is an organized storage used to keep data.

1.1 Definition

Data Structures

```
data = [1, 2, 3]
this = ['abc', 123]
grocery_list = ["eggs", "milk", "bread"]
```

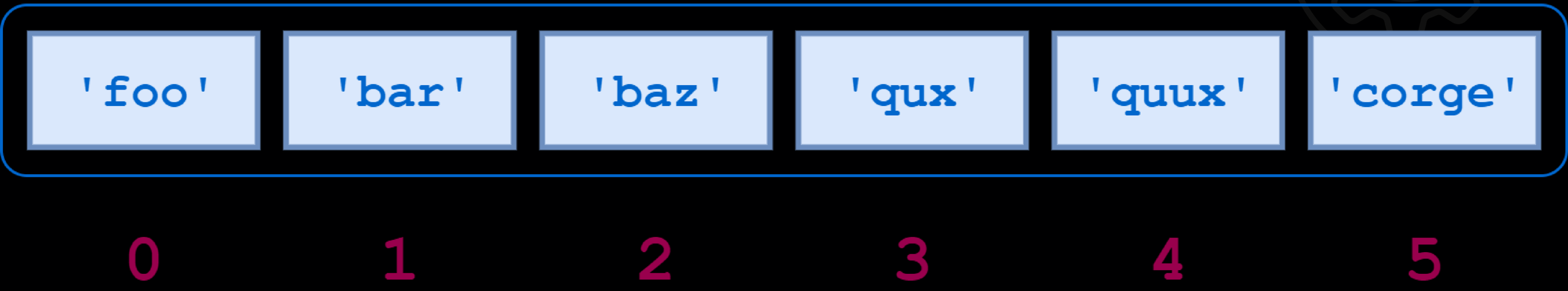
1.2 Types


Python Data Structures
Primitive
Float
Integer
String
Boolean
Non-Primitive
Built-In
List
Tuple
Dictionary
Set
User-Defined
Stack
Queue
LinkedList
Tree

1.3 Index

- An index refers to a position within an ordered list
- Think of it as some sort of location in a map
- For the built-in linear data structures, think of it like how you number your quiz answers.

1.3 Index



- Always starts with zero(0)
- Why?

Lists

- It has an order (index 0 to n)
- May be a mixture of data types
- items can be updated/overwritten

Syntax when Creating

<list_name> = [<content1>, <content2>]

Syntax when referencing Item

<list_name>[<index>]

2.1 List

Example

```
example = [1, 'two', 'three', 4, 5]
print(example)
print(example[1])
```

2.1 List

Example

```
[1, 'two', 'three', 4, 5]
two
```

2.1 List

Example

```
example = [1, 'two', 'three', 4, 5]
print(example)
example[1] = 5
print(example[1])
```

2.1 List

Example

```
[1, 'two', 'three', 4, 5]
5
```

Conclusion

- Lists are created the same way as variables, just with square brackets
- You can change the values of anything in the list
- We use indexes to refer to items in the list

Tuple

- It has an order (index 0 to n)
- May be a mixture of data types
- items, once tuple is created, cannot be updated/overwritten (immutable)

Syntax when Creating

<tuple_name> = (<content1>, <content2>)

Syntax when referencing Item

<tuple_name>[<index>]

Example

```
my_tuple = ("one", 2, 3, "four", 5)
print(my_tuple)
print(my_tuple[4])
```

Example

```
('one', 2, 3, 'four', 5)
5
```

Example

```
my_tuple = ("one", 2, 3, "four", 5)
print(my_tuple)
my_tuple[4] = 11
print(my_tuple[4])
```

Example

```
Traceback (most recent call last):
  File "C:/Users/Dell/Desktop/MyPy
thonPrograms/test6.py", line 3, in
<module>
    my_tuple[4] = 11
TypeError: 'tuple' object does not
support item assignment
```

Conclusion

- Tuples are created the same way as variables, just with parentheses
- You cannot change the values of anything in the tuple
- We use indexes to refer to items in the tuple

Set

- It has no order
- May be a mixture of data types
- There can be no duplicate values, all items are unique
- Items can be overwritten/updated

2.3 Set

Syntax when Creating

<set_name> = set([<val1>, <val2>])

2.3 Set

Example

```
my_set = set(["one", 2, 3, "four", 5])
print(my_set)
```

2.3 Set

Example

```
{2, 3, 5, 'one', 'four'}
```

2.3 Set

Example

```
my_set = set(["one", 2, 3, "four", 5])
print(my_set)
my_set[4] = 11
print(my_set)
```

Example

```
Traceback (most recent call last):
  File "C:/Users/Dell/Desktop/MyPy
thonPrograms/test6.py", line 3, in
<module>
    my_set[4] = 11
TypeError: 'set' object does not s
upport item assignment
```

2.3 Set

Example

```
my_set = set(["one", 2, 3, "one", 5])
print(my_set)
```

2.3 Set

Example

Conclusion

- presenting sets are done as a whole, they have no indexes assigned
- sets sort themselves
- sets unify duplicate values

Dictionary

- It is an unordered collection of data values (no indices)
- Dictionary holds the key : value pair.
- can use varied data types as key-value pairs

Syntax when Creating

<dictionary_name> = {<key1>: <value1>, <key2>: <value2>}

Syntax when referencing Item

<dictionary_name>[<key>]

2.4 Dictionary

Example

```
my_dictionary = {'manila': 'Philippines', 1:['hollywood', 3]}
print(my_dictionary['manila'])
```

2.4 Dictionary

Example

2.4 Dictionary

Example

```
my_dictionary = {'manila': 'Philippines', 1:['hollywood', 3]}
print(my_dictionary[1])
```

2.4 Dictionary

Example

```
['hollywood', 3]
```

Conclusion

- key-value pairs matter
- we may reference a key-value pair only by using the key
- dictionaries use curly braces

? Your Turn

Review Table

- fill in the following table in regards to the characteristics of the discussed Built-in Python Data Structures

? Your Turn

Review Table

- fill in the following table in regards to the characteristics of the discussed Built-in Python Data Structures

? Your Turn

Python	List	Tuple	Set	Dictionary
Ordered				
Mutable				
Values/Items can be duplicated				