Events

Recap session 3 - Timestamps

```
timestamp_seq = [ 0, 1, 2, 3]
ts = timestamp_seq.pop(0)
```

```
    …
    // retrieve current_ts
    …
    if current_ts >= ts:
        sample.play()
        ts = timestamp_seq.pop(0)
    time.sleep(0.001)
    …
    // repeat the above
```

Recap Session 3 - Multiple samples

```
# first item in sublist is the timestamp, second is
the sample index

event_seq = [[0, 0], [0.5, 1], [1.5, 0], [3.0, 1]]

event = event_seq.pop(0)
event[0] → timestamp
event[1] → bevat the sample index


# dictionary = duidelijk
```

Datastructuren

- list
- ...?

Datastructuren

- list
- tuple
- dictionary
- set
- queue
- stack
- ...

Datastructuren

- list

```
aList = [ 0, 1, 2, 3]
# ordered collection, similar to array
# mutable
# elements of various types are allowed
```

- tuple
- dictionary
- set
- queue
- stack
- ...

Datastructuren

- list
- tuple

```
aTuple = (‘foo’, ‘bar’)
# ordered collection, similar to a list
# immutable
# elements of various types are allowed
```

- dictionary
- set
- queue
- stack
- ...

Datastructuren

- list
- tuple
- dictionary

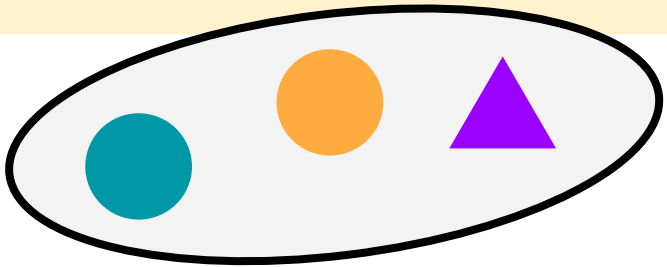
```
aDictonary = {‘sample’: ‘kick’, ‘ts’: 1.75}
# an associative array with key-value pairs
# mutable
# elements of various types are allowed
```

- set
- queue
- stack
- …

Datastructuren

- list
- tuple
- dictionary
- set

```
aSet = set([64, 62, ‘c’, 67, ‘g’])
# unordered collection
# mutable
# elements of various types are allowed
# no duplicates
```



- queue
- stack
- …

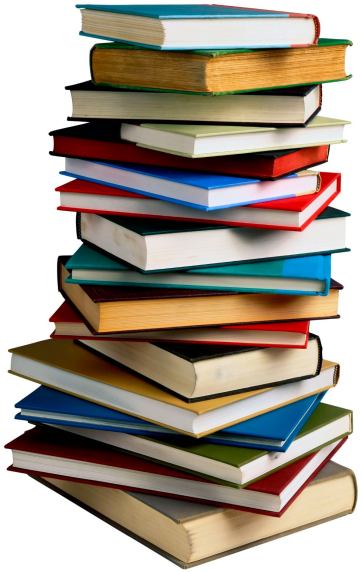
Datastructuren

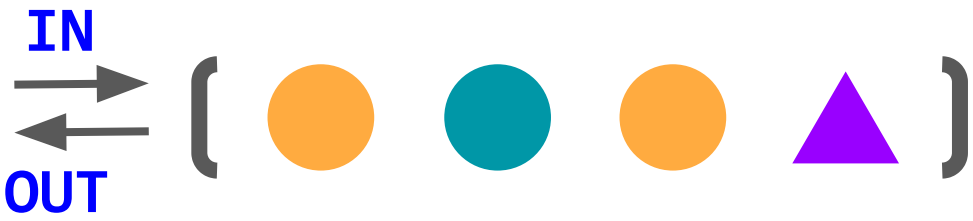
- list
- tuple
- dictionary
- set
- queue

- stack
- ...

Datastructuren

- list
- tuple
- dictionary
- set
- queue
- **stack**
- ...





Datastructuren

- list
- tuple
- dictionary
- set
- queue
- stack
- frozenset
- numpy array
- bytearray
- Counter
- OrderedDict
- String
- DefaultDict
- deque
- UserDict
- UserList
- UserString
- linked list
- trees
- ...

Datastructuren

- **list**
- tuple
- **dictionary**
- **set**
- **queue**
- **stack**
- frozenset
- **numpy array**
- bytearray
- Counter
- OrderedDict
- String
- DefaultDict
- deque
- UserDict
- UserList
- UserString
- **linked list**
- **trees**
- ...

Datastructuren

- **list**
- tuple
- **dictionary**
- **set**
- **queue**
- **stack**
- frozenset
- **numpy array**
- bytearray
- Counter
- OrderedDict
- String
- DefaultDict
- deque
- UserDict
- UserList
- UserString
- **linked list**
- **trees**
- ...

https://www.geeksforgeeks.org/python-data-structures/

Datastructuren nodig voor de eindopdracht

- **list**
- tuple
- **dictionary**
- set
- queue
- stack
- frozenset
- numpy array
- bytearray
- Counter
- OrderedDict
- String
- DefaultDict
- deque
- UserDict
- UserList
- UserString
- linked list
- trees
- ...

Dictionary

```
aDictonary = {‘sample’: ‘kick’, ‘ts’: 1.75}
# an associative array with key-value pairs
# mutable
# elements of various types are allowed
```

Dictionary - toepassing voor eindopdracht

```
# store the sample objects in a dictionary
samples = {
    ‘kick’:  sa.WaveObject.from_wave_file("../assets/Kick.wav"),
    ‘snare’:  sa.WaveObject.from_wave_file("../assets/Snare.wav"),
    ‘hihat’:  sa.WaveObject.from_wave_file("../assets/Hihat.wav")
}

# example of one event
timestamp = {‘sample’: ‘kick’, ‘ts’: 0.75}
```

Dictionary - eindopdracht

```
# store the sample objects in a dictionary
samples = {
    ‘kick’:  sa.WaveObject.from_wave_file("../assets/Kick.wav"),
    ‘snare’:  sa.WaveObject.from_wave_file("../assets/Snare.wav"),
    ‘hihat’:  sa.WaveObject.from_wave_file("../assets/Hihat.wav")
}

# example of one event
timestamp = {‘sample’: ‘kick’, ‘ts’: 0.75}

# RHYTHM GENERATION
# Generate lists with events (durations → timestamps), one list per sample
# Merge separate lists into one new list
# Orden the dictionaries in this merged new list according their timestamp
```

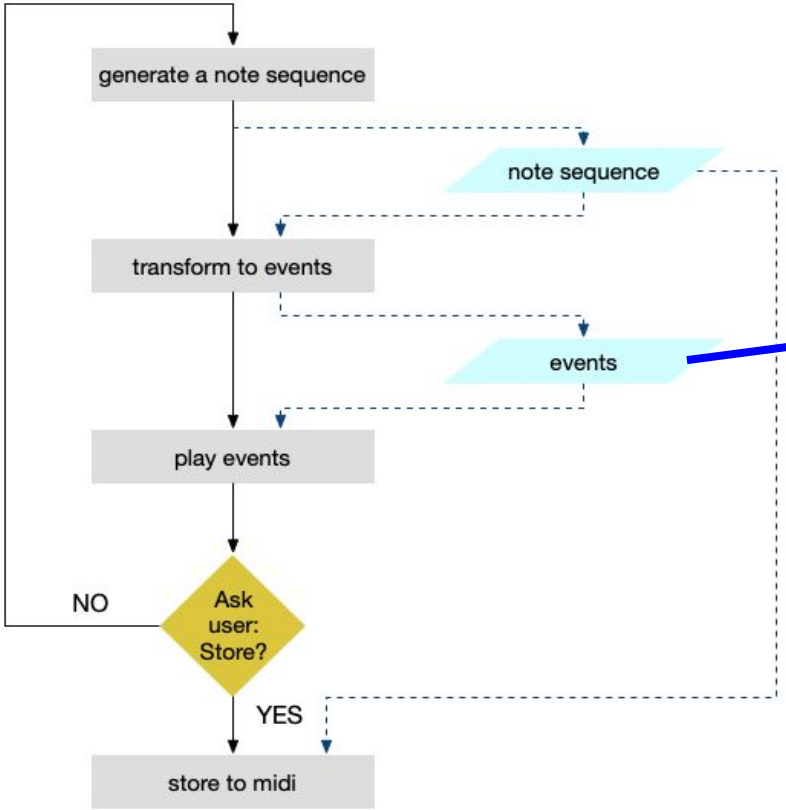
Dictionary - eindopdracht

```
# store the sample objects in a dictionary
samples = {
    ‘kick’:  sa.WaveObject.from_wave_file("../assets/Kick.wav"),
    ‘snare’:  sa.WaveObject.from_wave_file("../assets/Snare.wav"),
    ‘hihat’:  sa.WaveObject.from_wave_file("../assets/Hihat.wav")
}

# example of one event  - DO YOU ALSO WANT TO STORE DURATION??
timestamp = {‘sample’: ‘kick’, ‘ts’: 0.75}

# RHYTHM GENERATION
# Generate lists with events (durations → timestamps), one list per sample
# Merge separate lists into one new list
# Order the dictionaries in this merged new list according their timestamp
```

Dictionary - eindopdracht



DO YOU ALSO WANT TO STORE DURATION??

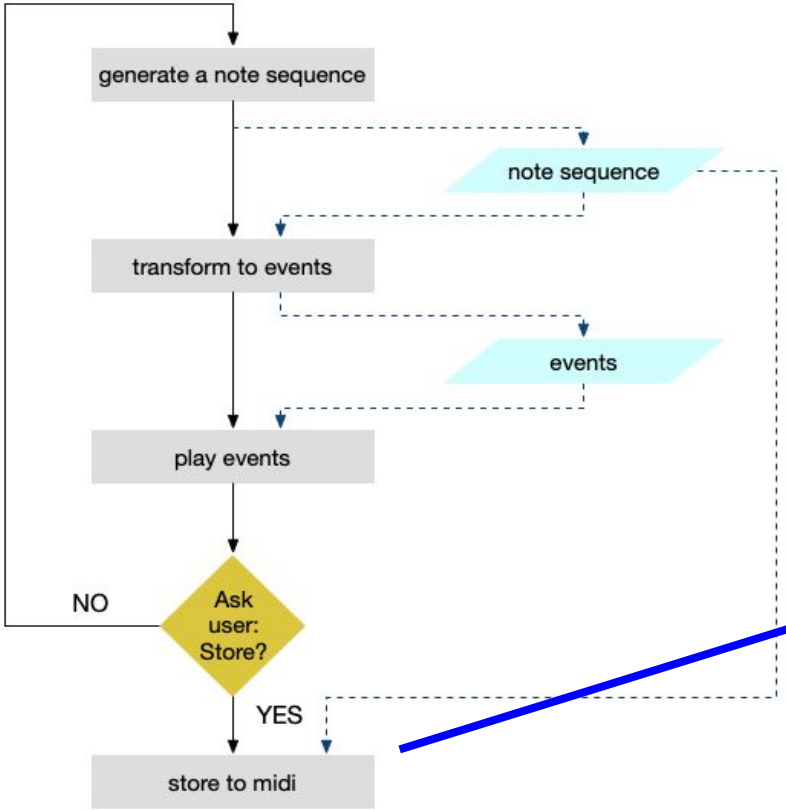
Dictionary - eindopdracht

```
# store the sample objects in a dictionary
samples = {
    ‘kick’:  sa.WaveObject.from_wave_file("../assets/Kick.wav"),
    ‘snare’:  sa.WaveObject.from_wave_file("../assets/Snare.wav"),
    ‘hihat’:  sa.WaveObject.from_wave_file("../assets/Hihat.wav")
}

# example of one event
timestamp = {‘sample’: ‘kick’, ‘ts’: 0.75}

# RHYTHM GENERATION
# Generate lists with events (durations → timestamps), one list per sample
# Merge separate lists into one new list
# Order the dictionaries in this merged new list according their timestamp
```

Dictionary - eindopdracht



separate tracks?

Dictionary - eindopdracht

```
# Order the dictionaries in this merged new list according their timestamp
```

sorting a list of dictionaries:

- manually
- with the sort() function