

# WEEK-3 REVIEW

- tuples
- lists
- dictionaries

# TUPLES

# TUPLES

```
t1 = ()
```

```
t2 = (1,'two',3)
```

# TUPLES

```
t1 = ()
```

```
t2 = (1,'two',3)
```

```
$ t1 =(1,'two',3)
```

```
$ t2 = (t1,3.25)
```

```
$ t1,(4,5,6)
```

```
$ t1+t2
```

```
$ (t1 + t2)[3]
```

```
$ (t1+t2)[2:5]
```

# TUPLES

```
t1 = ()
```

```
t2 = (1,'two',3)
```

```
$ t1 =(1,'two',3)
```

```
$ t2 = (t1,3.25)
```

```
$ t1,(4,5,6)
```

?

```
$ t1+t2
```

```
$ (t1 + t2)[3]
```

```
$ (t1+t2)[2:5]
```

# TUPLES

```
t1 = ()
```

```
t2 = (1,'two',3)
```

```
$ t1 =(1,'two',3)
```

```
$ t2 = (t1,3.25)
```

```
$ t1,(4,5,6)
```

```
$ t1+t2
```

?

```
$ (t1 + t2,15)
```

```
$ (t1+t2)[2:5]
```

# TUPLES

```
t1 = ()
```

```
t2 = (1, 'two, 3')
```

TUPLES ARE  
IMMUTABLE



# TUPLES

```
t1 = ()
```

```
t2 = (1, 'two', 3)
```

TUPLES ARE  
IMMUTABLE

EXAMPLE (FIND DIVISORS)

# LISTS

- mutable
- very much like strings
- empty list is written as []
- lists can contain lists inside
  - think them as arrays in Java and PHP

# LISTS (VISUALIZATION -1)

# LISTS (VISUALIZATION -1)

Techs → ['MIT', 'CALTECH']

Ivys → ['HARVARD', 'YALE', 'BROWN']

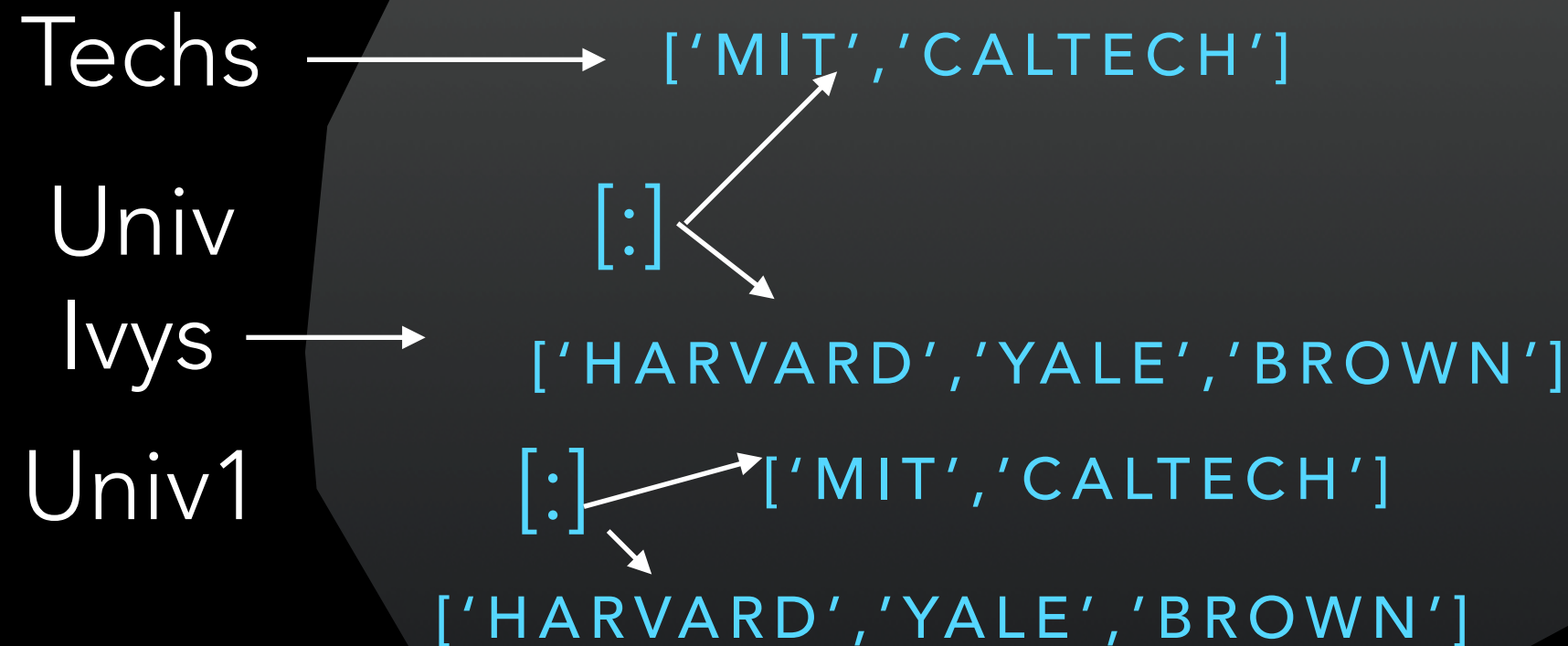
# LISTS (VISUALIZATION - 2)

```
[:]
```

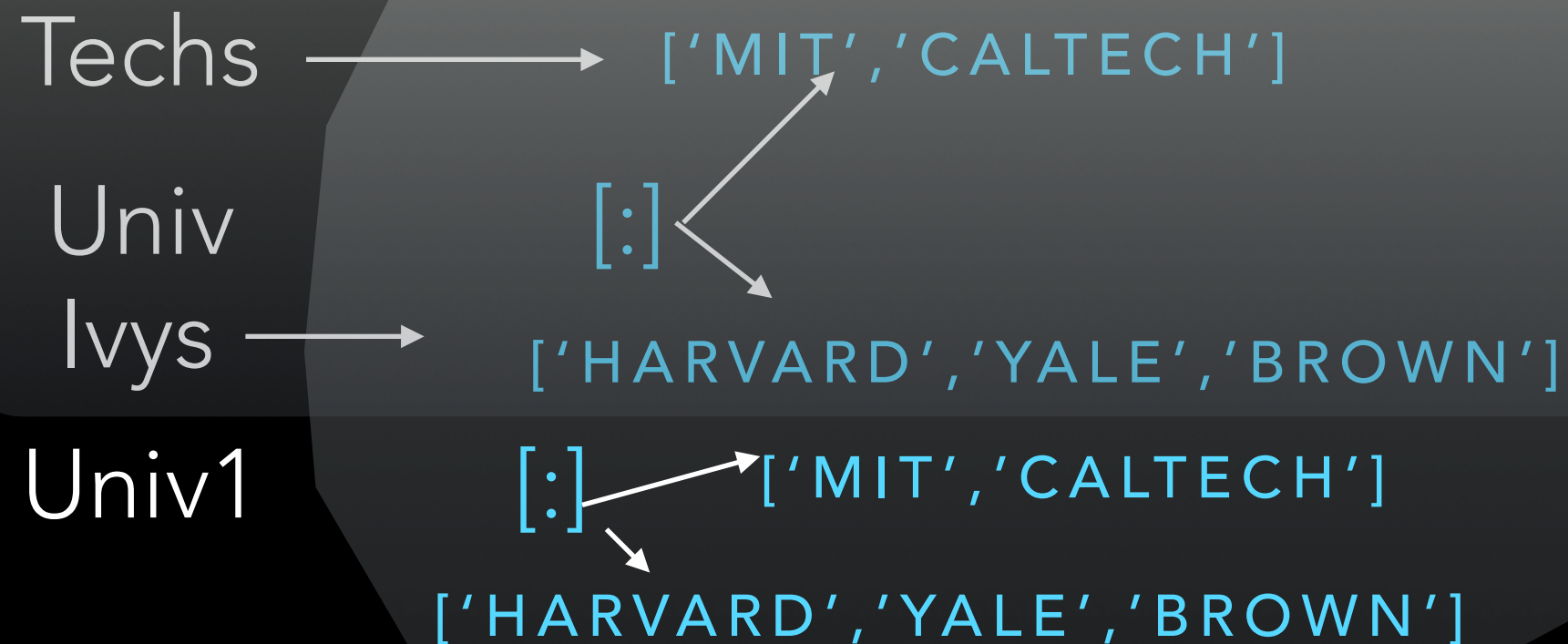
```
['HARVARD','YALE','BROWN']
```

```
[:]      ['MIT','CALTECH']
```

# LISTS (VISUALIZATION - 2)



# LISTS (VISUALIZATION - 2)



# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()



# LIST METHOD

L.append(e)

DISCUSSION

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

DISCUSSION

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

DISCUSSION

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

DISCUSSION

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

DISCUSSION

L.index(e)

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

DISCUSSION

L.pop(i)

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i) 

L.sort()

L.reverse()

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

DISCUSSION

L.reverse()



# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

DISCUSSION

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

DISCUSSION

# LIST METHOD

L.append(e)

L.count(e)

L.insert(i,e)

L.extend(L1)

L.remove(e)

L.index(e)

L.pop(i)

L.sort()

L.reverse()

DISCUSSION

CLONE LIST WHILE  
CHANGING VALUES OF  
LISTS IN LOOP

# STRING, TUPLES AND LISTS


- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

# STRING, TUPLES AND LISTS

## DISCUSSION

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`  **DISCUSSION**
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is True ..
- `s not in seq` is True ..
- `for e in seq`

# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

DISCUSSION

# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

DISCUSSION



# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is True ..
- `s not in seq` is True ..
- `for e in seq`

DISCUSSION

# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is True ..
- `s not in seq` is True ..
- `for e in seq`

DISCUSSION

# STRING, TUPLES AND LISTS

- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

DISCUSSION

# STRING, TUPLES AND LISTS

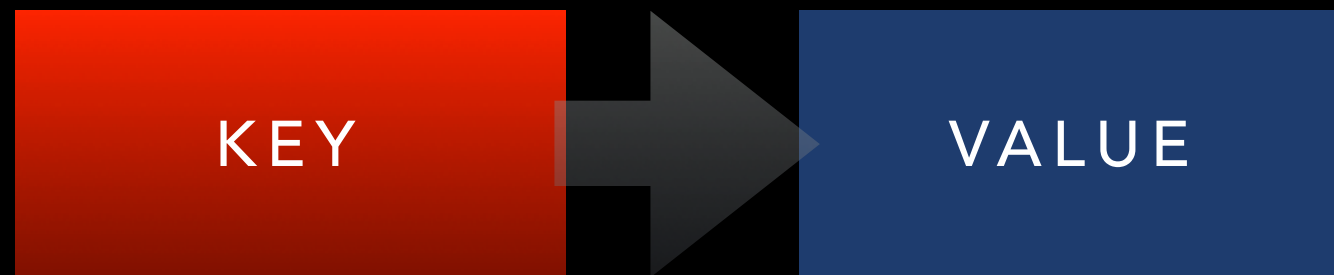
- `seq[i]`
- `len(seq)`
- `seq1 + seq2`
- `seq[start:end]`
- `e in seq` is `True` ..
- `s not in seq` is `True` ..
- `for e in seq`

DISCUSSION

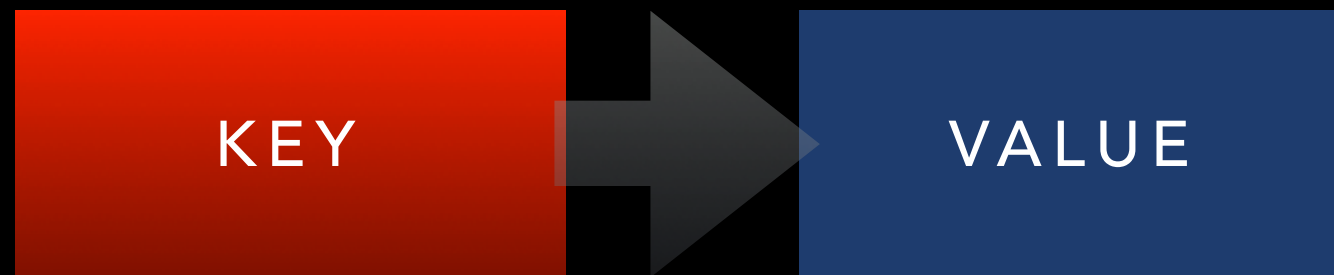
# LISTS OR TUPLES ??

- Advantages
- disadvantages

# DICTIONARIES

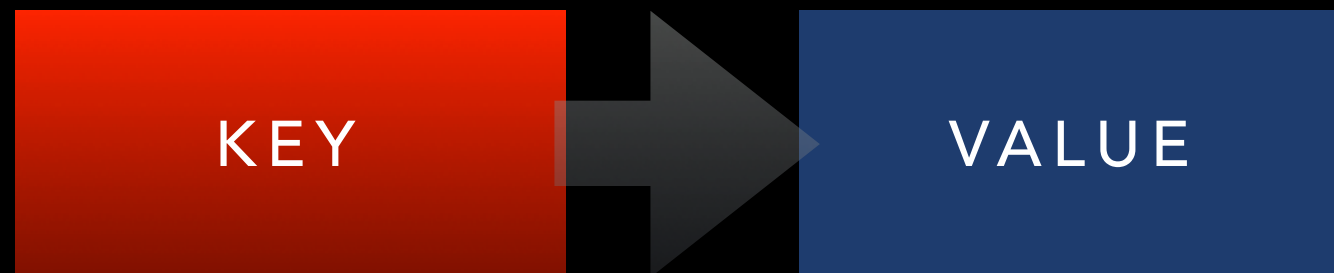


# DICTIONARIES



- Entries in Dictionaries are not indexed

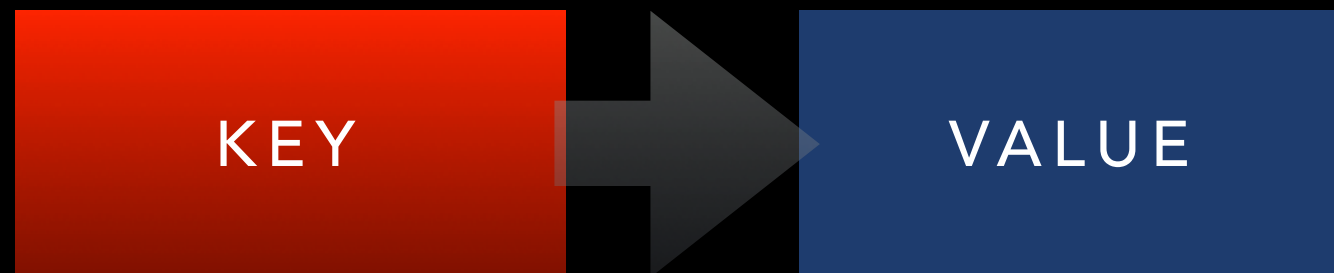
# DICTIONARIES



- Entries in Dictionaries are not indexed
- `exampleDict.keys()` will return list of keys



# DICTIONARIES



- Entries in Dictionaries are not indexed
- `exampleDict.keys()` will return list of keys
- for statement on dict will give **keys** not values

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

# HELPFUL METHOD - DICTIONARIES

## DISCUSSION

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

DISCUSSION

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

DISCUSSION

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`



DISCUSSION

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`



DISCUSSION

# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

DISCUSSION



# HELPFUL METHOD - DICTIONARIES

- `lend(d)`
- `d.keys()`
- `d.values()`
- `k in d`
- `d.get(k,v)`
- `del d[k]`
- `for k in d`

DISCUSSION