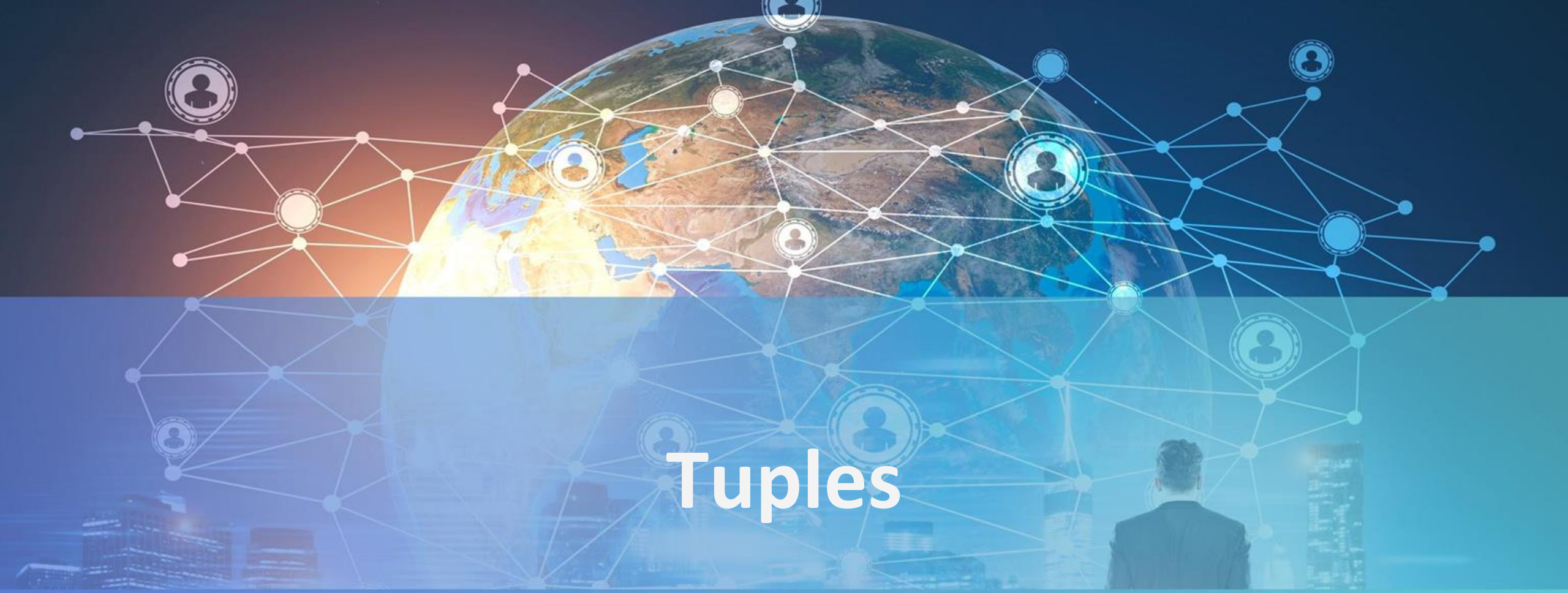


Tuples

GITAA
Transforming careers

In this lecture

- Tuples
  - Create tuples
  - Indexing
  - Access components
  - Slicing
  - Built in functions
  - Combine multiple tuples
  - Modify components

Tuples

Tuples

- Consists of an ordered collection of objects
- Some of the operations on tuples are similar to lists
- Tuples are enclosed between parentheses ( )
- Immutable
- Once created they cannot be modified

Creating tuples

- Create a tuple with employee id, name, age, salary details

```
employee_details =('P001','John',35,40000)
```

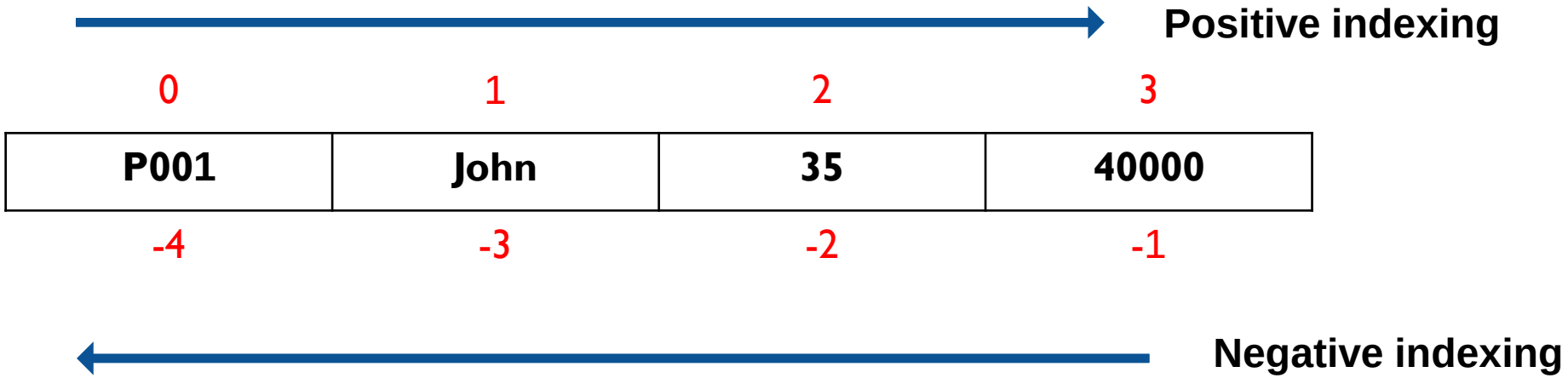
- Print the tuple

```
In [2]: print(employee_details)
('P001', 'John', 35, 40000)
```

Indexing

- Consider the following tuple

```
In [2]: print(employee_details)
('P001', 'John', 35, 40000)
```



Accessing components of a tuple

- To access components, use single slicing operator '[ ]'
- Syntax: `tuple_name[index]`
- To extract ***id*** from ***employee_details***

```
In [3]: print(employee_details[0])
P001
```

Accessing components of a tuple

- Extract ***salary*** from the ***employee_details***

```
In [4]: print(employee_details[3])
40000
```

- Index number greater than 3 will be the out of range

```
In [5]: print(employee_details[5])
Traceback (most recent call last):

  File "<ipython-input-5-bc7184be9c96>", line 1, in <module>
    print(employee_details[5])

IndexError: tuple index out of range
```

Slicing

- Used to access a set of elements from a tuple by creating a range of index numbers ***[x:y]***
- x- index number is where the slice starts **(inclusive)**
- y- index number is where the slice ends **(exclusive)**
- Elements are extracted from x to y-1

Slicing

- To extract ***name & age*** from the ***employee_details***

```
In [11]: print(employee_details[1:3])
('John', 35)
```

- Only elements with index number 1 and 2 will be printed

Slicing

- To extract ***id, employee name, age*** from the ***employee_details***

```
In [12]: print(employee_details[:3])
('P001', 'John', 35)
```

- Here all elements are printed except the one corresponding to index number 3

Length of a tuple

- `len()` - returns the length of a tuple
- Syntax: `len(tuple_name)`

```
In [13]: len(employee_details)
Out[13]: 4
```

Finding minimum & maximum from tuple

- `min()` - returns the lowest value in the tuple
- `max()` - returns the highest value in the tuple
- Syntax: `min(tuple_name)`
- Create a tuple of marks secured by students in English

```
In [14]: english= (56,85,96,75,12)

In [15]: min(english)
Out[15]: 12

In [16]: max(english)
Out[16]: 96
```

Combining two tuples

- Two tuples can be concatenated as follows `(tuple1)+(tuple2)`
- Create a tuple with employee education and department details

```
employee_details2 = ('M.Com','Accountants')
```

Combining two tuples

```
employee_details =('P001','John',35,40000)
```
→ **Tuple 1**

```
employee_details2 = ('M.Com','Accounts')
```
→ **Tuple 2**

- Print the updated tuple

```
In [12]: print(employee_details+employee_details2)
('P001', 'John', 35, 40000, 'M.Com', 'Accounts')
```

Modifying components of a tuple

- Tuples are different from lists in the sense tuples cannot be modified
- Elements cannot be added or removed from tuples using index number or functions (append, del, remove etc.)

```
In [17]: employee_details[0]='P002'
Traceback (most recent call last):

  File "<ipython-input-17-ea7bbb0815e1>", line 1, in <module>
    employee_details[0]='P002'

TypeError: 'tuple' object does not support item assignment
```

Summary

- Create tuples
- Indexing
- Accessing components from a tuple
- Built in functions- len( ) , min( ), max( )
- Concatenation of tuple
- Modifying components


operation == "MIRROR_X":
irror_mod.use_x = True
irror_mod.use_y = False
irror_mod.use_z = False
operation == "MIRROR_Y":
irror_mod.use_x = False
irror_mod.use_y = True
irror_mod.use_z = False
operation == "MIRROR_Z":
irror_mod.use_x = False
irror_mod.use_y = False
irror_mod.use_z = True
selection at the end -add
ob.select= 1
er_ob.select=1
ntext.scene.objects.acti
("Selected" + str(modifier
irror_ob.select = 0
bpy.context.selected_ob
ata.objects[one.name].se
int("please select exact

THANK YOU