Introduction to Computer Programming

Week 6.1: Importing Python files

In-class Demos

Try it yourself

Example 1a:

Create the file structure shown below within a new folder called `lecture_6` .

Add the content shown within each file.

```
lecture_6/
│
├── __init__.py
├── capitals.py
└── main.py
```

`__init__.py`

```
# (empty file)
```

`capitals.py`

```
Japan = ('Japan', 'Tokyo')
Germany = ('Germany', 'Berlin')
```

`main.py`

```
# (empty file)
```

Example 1b:

Within `main.py` , print the output below:
`The capital of Japan is Tokyo`

Solution: Example 1b

main.py

```
import capitals

print(f'The capital of {capitals.Japan[0]} is {capitals.Jap
an[1]}')
```

Try it yourself

Example 1c:

Within `main.py` , print the output below:

```
The capital of Germany begins with B
```

Example 1d: Can you make the code in `main.py` any more concise?

Solution: Example 1c

main.py

```
import capitals

print(f'The capital of {capitals.Germany[0]} begins with {c
apitals.Germany[1][0]}')
```

Solution: Example 1d

Changing the module name in the local namespace

main.py

```
import capitals as c

print(f'The capital of {c.Japan[0]} is {c.Japan[1]}')

print(f'The capital of {c.Germany[0]} begins with {c.German
y[1][0]}')
```

Solution: Example 1d

Importing *individual items* from a module

Whatever comes after `import` is added to the local namespace

main.py

```
from capitals import Japan, Germany

print(f'The capital of {Japan[0]} is {Japan[1]}')

print(f'The capital of {Germany[0]} begins with {Germany[1]
[0]}')
```

Solution: Example 1d

Importing *individual items* from a module and renaming

Whatever comes after `import` is added to the local namespace

main.py

```
from capitals import Japan as J, Germany as G

print(f'The capital of {J[0]} is {J[1]}')

print(f'The capital of {G[0]} begins with {G[1][0]}')
```

In [ ]:

```
1
```