Using spaces around operators and commas

- Improves code readability
- Makes commands look spacious and discrete

```
a=b+c
k=function_1(a,b)
D=[1,2,3,4]
```

```
a = b + c
k = function_1(a, b)
D = [1, 2, 3, 4]
```

Using functions for blocks of codes

- Create separate functions for functionalities

```
…
if a>b:
    c = c + 5
else:
    c = c - 3
…
```

```
def function_1(a, b):
    if a > b:
        c = c + 5
    else:
        c = c - 3
    return c
…
c = function_1(a, b)
…
```

Recap

In this video, you learned that:

- Writing consistent code helps all team members read and understand the code easily
- The PEP8 guidelines for code readability include four spaces for indentation, use blank lines to separate functions and classes, and use spaces around operators and after commas
- Coding conventions for consistency and manageability include add larger blocks of code inside functions, name functions and files using lowercase with underscores, name classes using CamelCase, and name constants in capital letters with underscores separating words
- Use Static code analysis method to evaluate your code against a predefined style and standard without executing the code

Naming functions and files

- Use lowercase with underscores

Function naming convention

```
S=compSurfaceRadiation()

B=flux_airgap_SDM()
```

```
S=comp_surface_radiation()

B=comp_flux_airgap_SDM()
```

Using blank lines to separate functions and classes

```
def function_1():
    <statement 1>
    <statement 2>
class UserClass():
    <statement 1>
    <statement 2>
```

```
def function_1():
    <statement 1>
    <statement 2>

class UserClass():
    <statement 1>
    <statement 2>
```

Naming classes using CamelCase

- Coding convention well-accepted in the coding community
- Helps distinguish between classes and functions

```
class Lam_squirrel_Cage:
```

```
class LamSquirrelCage:
```

Naming constants

- Capitalize all words
- Separate the words with underscores

```
maxfile

Max_File_Upload_Size

maxfileuploadsize
```

```
MAX_FILE_UPLOAD_SIZE
```