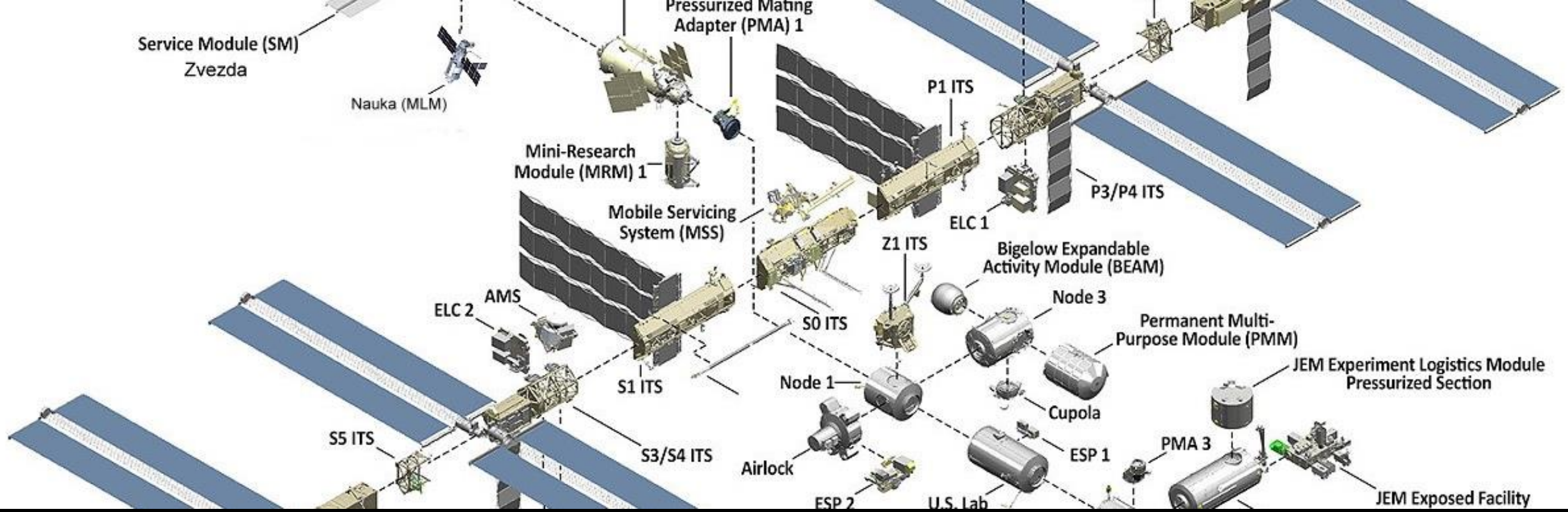
FUNDAMENTALS OF PROGRAMMING

Week 8 - Built-in modules of Python

Janne Parkkila, Dr.Sc. (tech)



MODULES

Modules, Libraries... and that’s it.

WHAT IS A MODULE?

- A module is a file containing Python code that defines functions, classes, and constants for use in other Python programs.
- This week, we consider several useful modules included with the Python installation.
- The Python standard library contains over 200 modules, although the exact number varies between distributions.

MODULES

- In Python, **modules** are files with .py –extensions
- Modules contain Python code that can be imported (=used) inside another Python program
- This way we can use code other people have written (foundation for Open Source!)
- Python has built-in modules, e.g. modules that come with Python installation. For example math & datetime
- There is a HUGE NUMBER of 3rd party Python modules, created by the community.
  - Some of them are really, really, really, absolutely amazing!
  - Some are nah
  - Some don’t even work
  - And some steal your information or are dangerous in other manners
  - More on this in later lectures.

HOW TO IMPORT A MODULE?

IMPORT *

- import module_name
- The simplest way to import a module.
- Allows you to access everything in the module by using the dot notation, such as module_name.function_name or module_name.variable_name
- For example, if you want to use the math module, you can write import math and then use math.pi or math.sqrt() in your code.

FROM MODULE IMPORT *

- from module_name import item_name
- Allows you to import a specific item from a module, such as a function, a class, or a variable.
- You can then use the item directly in your code without the dot notation.
- For example, if you want to use the pi constant from the math module, you can write:

```
from math import pi
```

and then use pi in your code.

IMPORT MODULE AS *

- from module_name import item_name as alias
- Allows you to import a specific item from a module and give it a shorter or different name.
- This can be useful if the item name is long or conflicts with another name in your code.
- Just don’t overdo this and generally a good practise is not to make up your own names for existing modules.
- There are some common conventions, like numpy = np, datetime = dt, tensorflow = tf. But you will see this in use later.

GETTING HELP

- You can get help on modules by writing help("module_name")
- Example: `help(“math”)`