

<http://modules.sourceforge.net/>

Makes **different versions** of one piece of software available.

Can **define** or **change** environment variables required for using a piece of software:

PATH

CPATH

LIBRARY_PATH

LD_LIBRARY_PATH

...



Use `module avail` to find installations of a package:

```
1 skylake trainee00@l44:~$ module avail intel-oneapi-mpi
2 zen trainee00@l55:~$ module avail amdblis
```

This is similar to `spack find`, and often shows the **same** packages.

Some software we installed manually so you can **only** load it with modules!

Package names are **case sensitive**, like **Matlab**, **Mathematica**, ...



Load your module

Use `module load` to load your package:

```
1 skylake trainee00@l44:~$ module load openmpi/4.1.4-gcc-12.2.0-xt53foa
```

Always state the whole line:

```
2 skylake trainee00@l44:~$ module load openmpi/4.1.4
3 ERROR: Unable to locate a modulefile for 'openmpi/4.1.4'
```

Module names differ depending on cluster (CPU architecture):

```
4 skylake trainee00@l44:~$ module load python/3.11.3-gcc-12.2.0-rtzvjk0
5 zen trainee00@l55:~$ module load python/3.11.3-gcc-12.2.0-hn7p65z
6 cuda-zen trainee00@l55:~$ module load python/3.9.15-gcc-12.2.0-dnctq7y
```



Type `module list` to show loaded modules:

```
1 zen trainee00@l55:~$ module list
2 zen trainee00@l55:~$ module list -t
3 Currently Loaded Modulefiles:
4 netlib-lapack/3.10.1-gcc-12.2.0-4qrxbdw
5 python/3.9.15-gcc-12.2.0-my6jxu2
6 py-setuptools/63.0.0-gcc-12.2.0-jru4czzh
7 py-numpy/1.24.3-gcc-12.2.0-muackhh
8 ...
```



Type `module purge` to remove all loaded modules:

```
1 module purge
```

Recommended in job scripts before loading any modules (reproducibility).

Use `module rm` or `module unload` to remove/unload a list of modules:

```
2 zen trainee00@l55:~$ module rm <modulename> <modulename> ...
```

```
3 zen trainee00@l55:~$ module unload <modulename> <modulename> ...
```



Use `module show` to show the variables added to your environment by a module:

```
1 zen trainee00@l155:~$ module show python/3.11.3-gcc-12.2.0-hn7p65z
2 -----
3 /opt/sw/zen/spack-0.19.0/share/spack/modules/linux-almalinux8-zen3...
4
5 module-whatis      {The Python programming language.}
6 prepend-path      PATH /gpfs/opt/sw/zen/spack-0.19.0/opt/spack/linux...
7 prepend-path      LIBRARY_PATH /gpfs/opt/sw/zen/spack-0.19.0/opt/spa...
8 prepend-path      CPATH /gpfs/opt/sw/zen/spack-0.19.0/opt/spack/linu...
9 prepend-path      MANPATH /gpfs/opt/sw/zen/spack-0.19.0/opt/spack/li...
10 ...
```

No need to load a module before inspecting it.



Sometimes a module also **needs** dependencies:

```
1 zen trainee00@l155:~$ module load py-numpy/1.24.3-gcc-12.2.0-muackhh
2 Loading py-numpy/1.24.3-gcc-12.2.0-muackhh
3   Loading requirement: netlib-lapack/3.10.1-gcc-12.2.0-4qrxbdw
4     python/3.9.15-gcc-12.2.0-my6jxu2
5     py-setuptools/63.0.0-gcc-12.2.0-jru4czh
```

Add `--auto` to automatically load all the dependencies:

```
6 zen trainee00@l155:~$ module load --auto py-numpy/1.24.3-gcc-12.2.0-muackhh
```

Write the line `export MODULES_AUTO_HANDLING=1` to your `~/.bashrc` to **always** load all dependencies, or never to with `export MODULES_AUTO_HANDLING=0`.



- login to VSC-4/5
- find a recent **python** module.
- load a recent **mathematica** module.
- load a recent **matlab** module.
- list all **loaded** modules.
- unload **all** modules.
- find a **matplotlib** *python* module.
- find an **openfoam** module compiled with **gcc**.
- find an **openfoam** module compiled with **intel**.
- what version of **numpy** does the module
`py-matplotlib/3.6.2-gcc-12.2.0-wbe7m7i` need.

