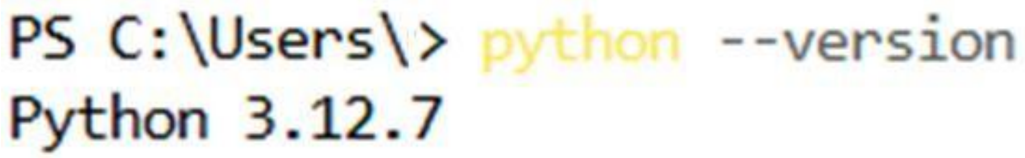
Practical 1

1.1	Date:

Aim: To develop environment configuration skills by setting up Python, Anaconda, and virtual environments for efficient project dependency management.

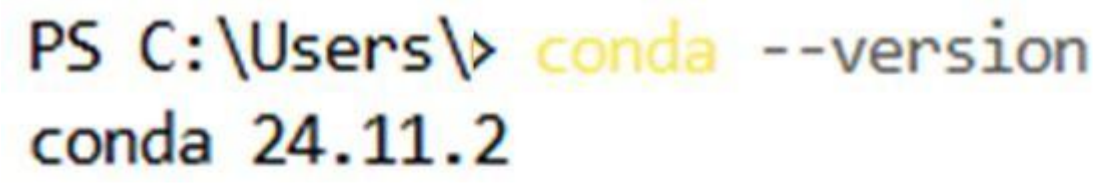
Code:
Installation

- Install Python and verify its installation using the command prompt or terminal.



```
PS C:\Users\> python --version
Python 3.12.7
```

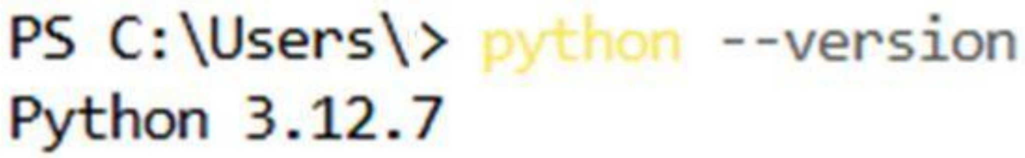
- Install Anaconda and validate it using the command prompt or terminal.



```
PS C:\Users\> conda --version
conda 24.11.2
```

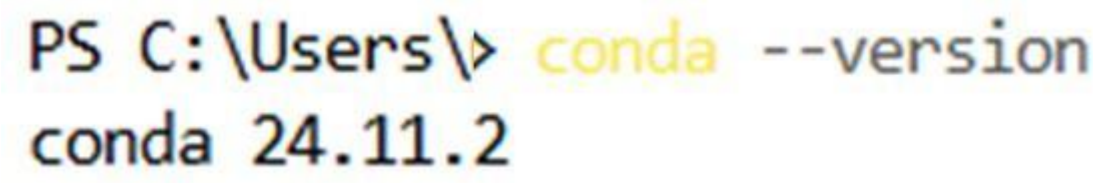
Configuration

- Set up environment variables (if required) to access Python and Conda from the command line.



```
PS C:\Users\> python --version
Python 3.12.7
```

- Verify the versions of Python and Anaconda installed on the system.

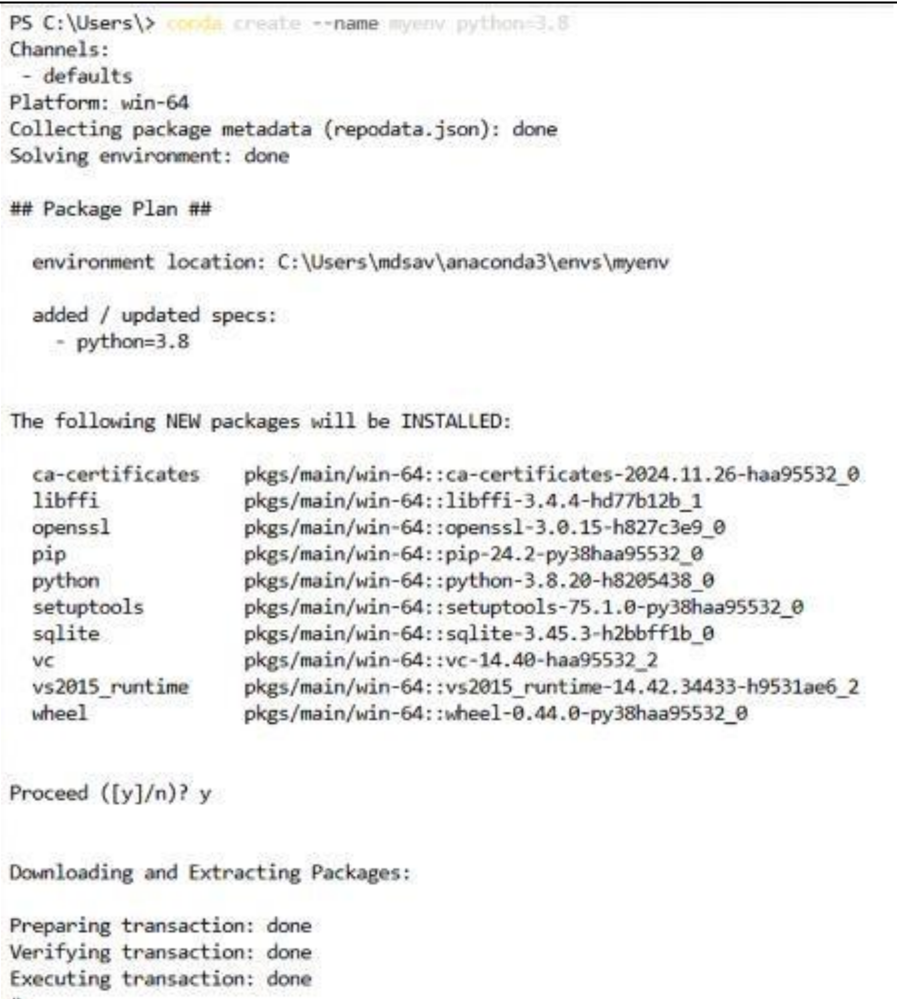


```
PS C:\Users\> conda --version
conda 24.11.2
```

Virtual Environment

- Create a new virtual environment using both Conda and venv methods.
- Activate and deactivate the created virtual environment.
- Install a specific package (like NumPy) in the virtual environment and verify its installation.
- List installed packages in the virtual environment.

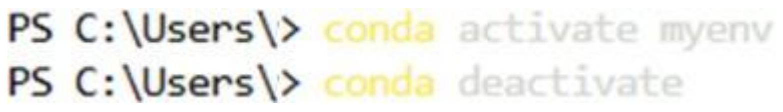
<u>FOR CONDA:</u>



```
PS C:\Users\> conda create --name myenv python=3.8
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\mdsav\anaconda3\envs\myenv

  added / updated specs:
    - python=3.8


The following NEW packages will be INSTALLED:

  ca-certificates    pkgs/main/win-64::ca-certificates-2024.11.26-haa95532_0
  libffi             pkgs/main/win-64::libffi-3.4.4-hd77b12b_1
  openssl            pkgs/main/win-64::openssl-3.0.15-h827c3e9_0
  pip                pkgs/main/win-64::pip-24.2-py38haa95532_0
  python             pkgs/main/win-64::python-3.8.20-h8205438_0
  setuptools         pkgs/main/win-64::setuptools-75.1.0-py38haa95532_0
  sqlite             pkgs/main/win-64::sqlite-3.45.3-h2bbff1b_0
  vc                 pkgs/main/win-64::vc-14.40-haa95532_2
  vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.42.34433-h9531ae6_2
  wheel              pkgs/main/win-64::wheel-0.44.0-py38haa95532_0


Proceed ([y]/n)? y


Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
```

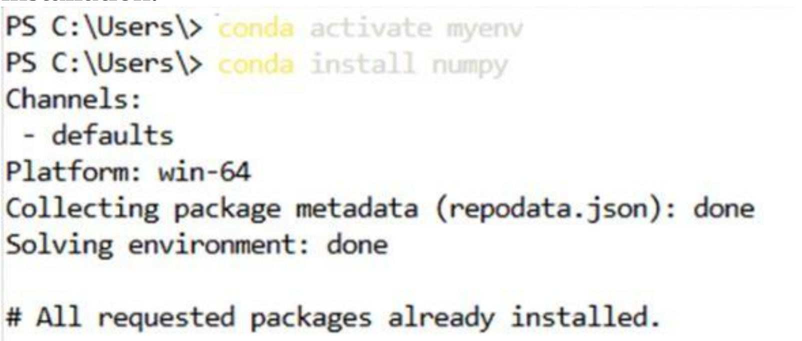
```
# To activate this environment, use
#
#     $ conda activate myenv
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Aim: Activate and deactivate the created virtual environment.

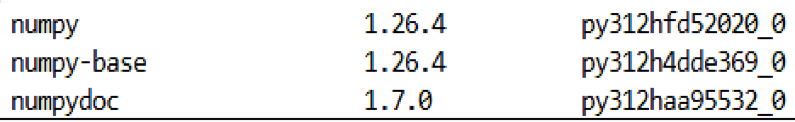


```
PS C:\Users\> conda activate myenv
PS C:\Users\> conda deactivate
```

Aim: Install a specific package (like NumPy) in the virtual environment and verify its installation.



```
PS C:\Users\> conda activate myenv
PS C:\Users\> conda install numpy
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

# All requested packages already installed.
```



```
numpy                     1.26.4          py312hfd52020_0
numpy-base                1.26.4          py312h4dde369_0
numpydoc                  1.7.0           py312haa95532_0
```

Conclusion/Summary:

Conclusion:
Setting up and configuring your Python and Anaconda environments, as well as effectively managing virtual environments, is a fundamental skill for efficient project dependency management. By completing the outlined steps, you ensure that you can work with different projects simultaneously without conflicts and maintain a clean and organized development environment.

Summary of the Process:

1. Installation: Installing Python and Anaconda and validating their installation ensures that you have the necessary tools for development.
2. Configuration: Setting up environment variables and verifying the versions of Python and Anaconda allows for smooth access and operation from the command line.
3. Virtual Environment: Creating, activating, and deactivating virtual environments using Conda and venv, along with managing packages within these environments, provides an isolated space for each project.
4. Project Isolation: Understanding the need for project isolation and the role of virtual environments in achieving this helps prevent dependency conflicts and ensures that each project can run with its specific requirements.

By mastering these skills, you enhance your ability to manage dependencies effectively, avoid conflicts between projects, and create a more efficient workflow. This not only boosts productivity but also contributes to better project organization and reliability. Keep practicing and refining these techniques to become proficient in environment configuration and management.

MITUL : 4-1-2024 **Student Signature & Date**	**Marks:**	**Evaluator Signature & Date**