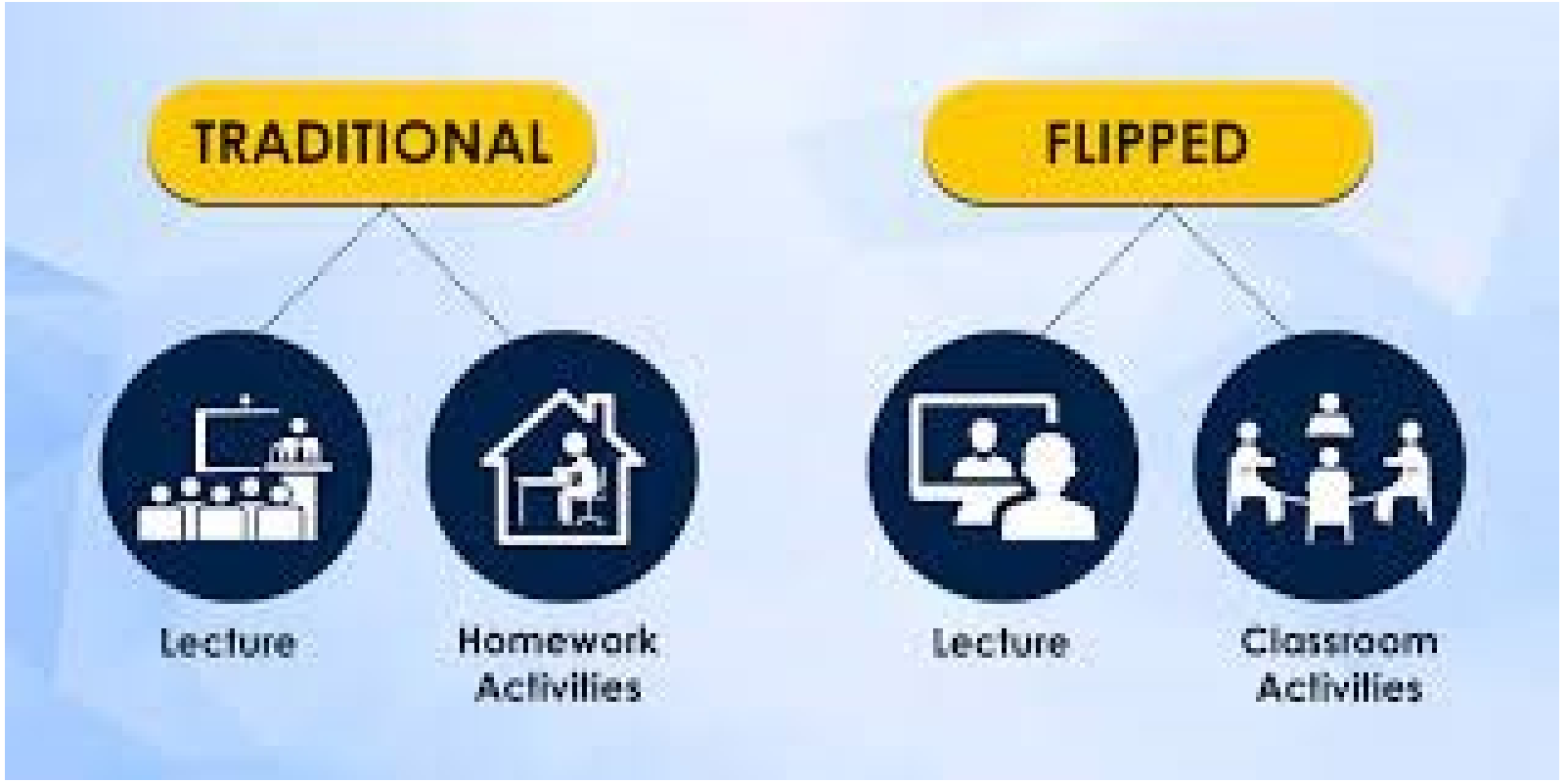


Lecture 9

File Compression

- File compression and decompression are essential techniques for efficient data storage and transfer.
- Various libraries in Python make it easy to work with compressed files.
- **zipfile, tarfile, and gzip.**

ZIP

- **Format**: ZIP is an archive file format that supports lossless data compression. It can contain multiple files and directories.
- **Compression**: The ZIP format allows for various compression algorithms, but DEFLATE is the most commonly used.
- **Usage**: ZIP files are widely used for their cross-platform compatibility.
- **Python Module**: zipfile

TAR

- **Format**: TAR (Tape Archive) is a format used to collect many files into one archive file, often called a tarball. TAR itself does not compress the files, it merely combines them.
- **Compression**: TAR archives are often compressed using Gzip or Bzip2 to reduce their size. The result is a .tar.gz or .tar.bz2 file.
- **Usage**: Commonly used on Unix and Linux systems for packaging software distributions.
- **Python Module**: tarfile

GZIP

- **Format**: GZIP (GNU Zip) is a file format and a software application used for file compression. Unlike ZIP, GZIP is used to compress a single file.
- **Compression**: GZIP uses the DEFLATE algorithm, which combines LZ77 and Huffman coding.
- **Usage**: Typically used in Unix and Linux environments to compress single files. Combined with TAR for archiving multiple files.
- **Python Module**: gzip

Compressing Files Using zipfile

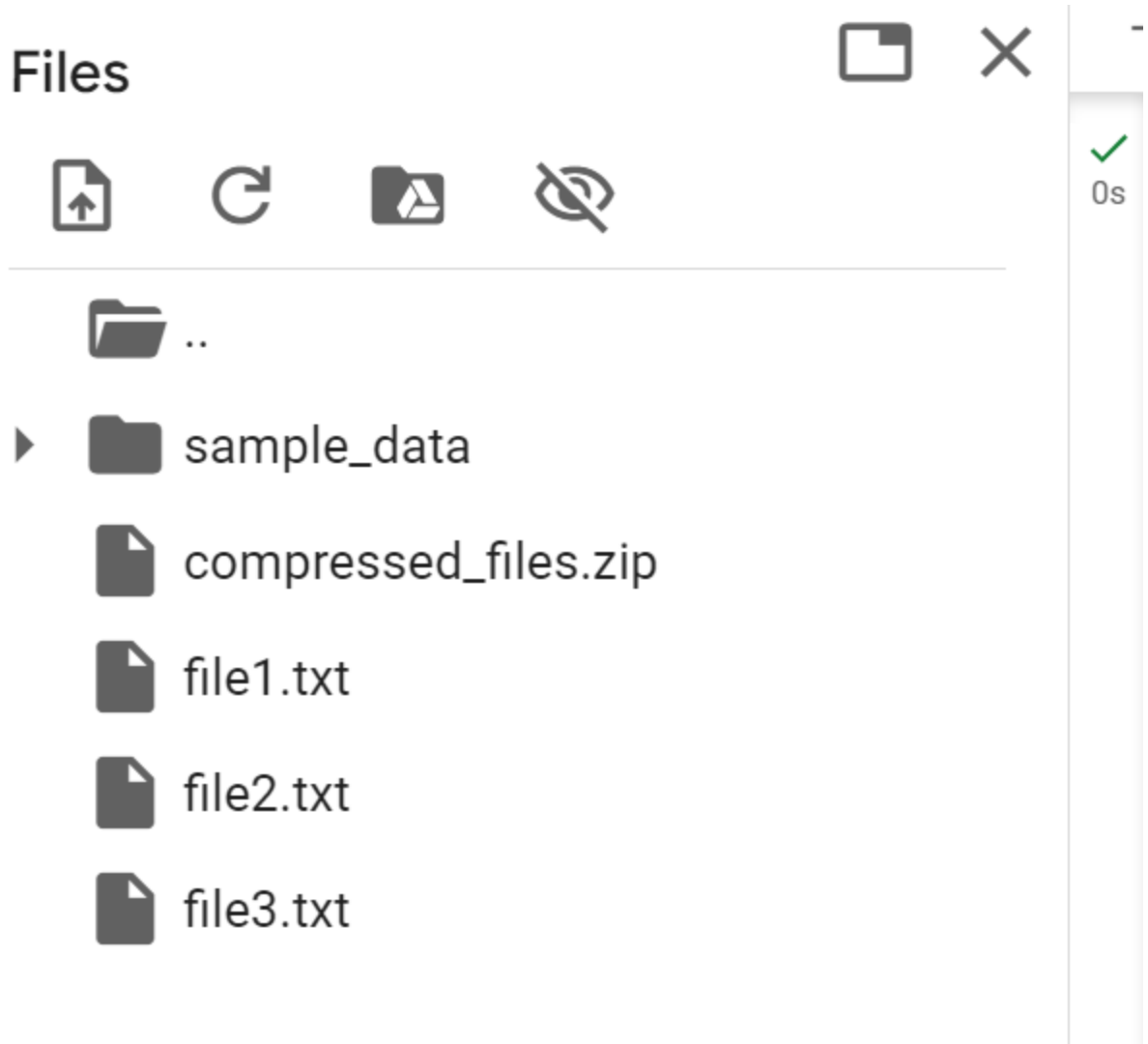
The zipfile module allows to create ZIP archives, which are a popular format for compressing multiple files.

```
import zipfile
import os

# Create dummy files for testing
dummy_files = ['file1.txt', 'file2.txt', 'file3.txt']
for file in dummy_files:
    with open(file, 'w') as f:
        f.write(f"This is a dummy file: {file}")

def compress_files_zip(files, output_zip):
    with zipfile.ZipFile(output_zip, 'w') as zipf:
        for file in files:
            if os.path.exists(file):
                zipf.write(file, compress_type=zipfile.ZIP_DEFLATED)
            else:
                print(f"File not found: {file}")

# Usage
files_to_compress = ['file1.txt', 'file2.txt', 'file3.txt']
output_zip = 'compressed_files.zip'
compress_files_zip(files_to_compress, output_zip)
```

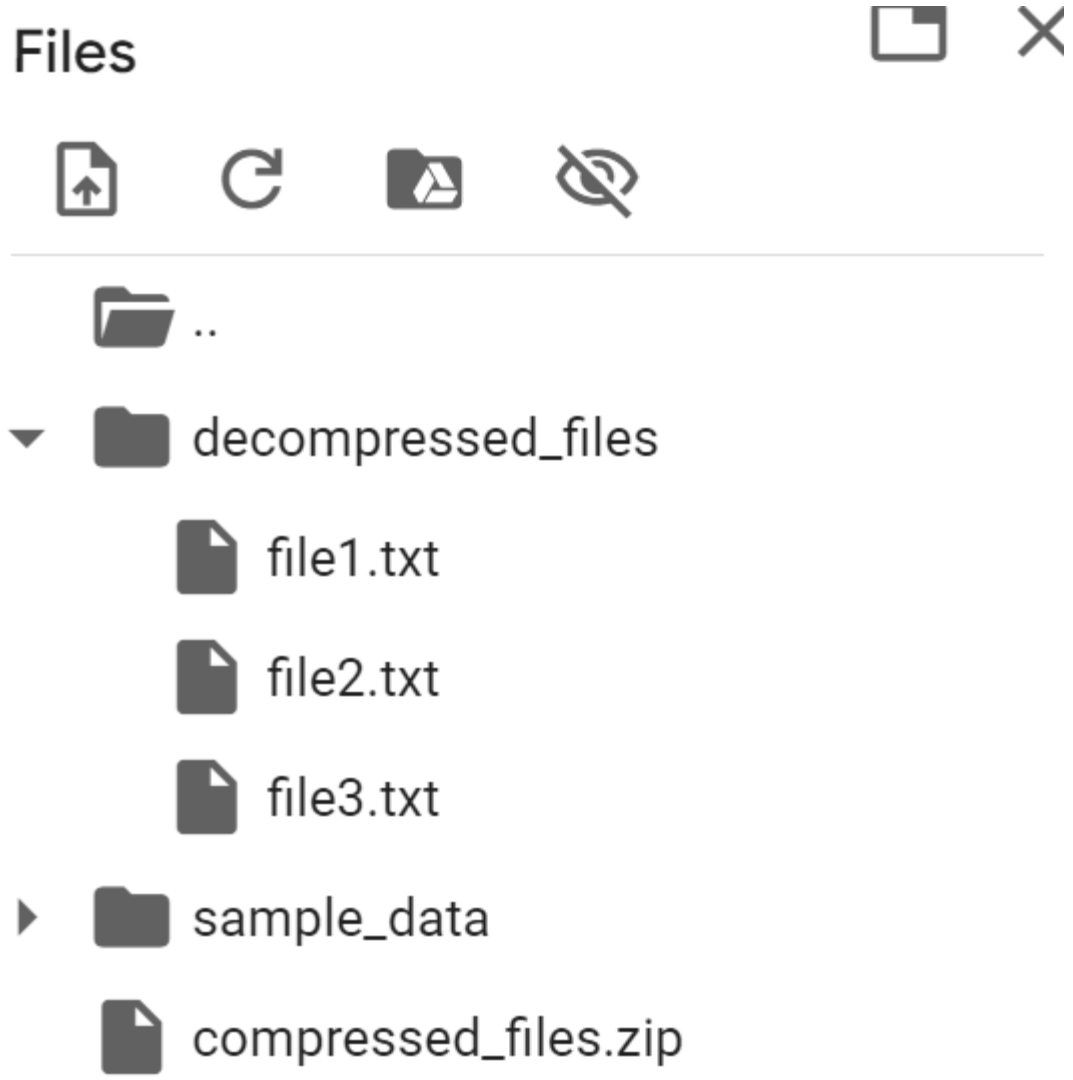

Files
..
sample_data
compressed_files.zip
file1.txt
file2.txt
file3.txt
0s

Decompressing Files Using zipfile

```
def decompress_zip(input_zip, output_dir):
    with zipfile.ZipFile(input_zip, 'r') as zipf:
        zipf.extractall(output_dir)

# Usage
input_zip = 'compressed_files.zip'
output_dir = './decompressed_files'
decompress_zip(input_zip, output_dir)
```

Decompressing Files Using zipfile



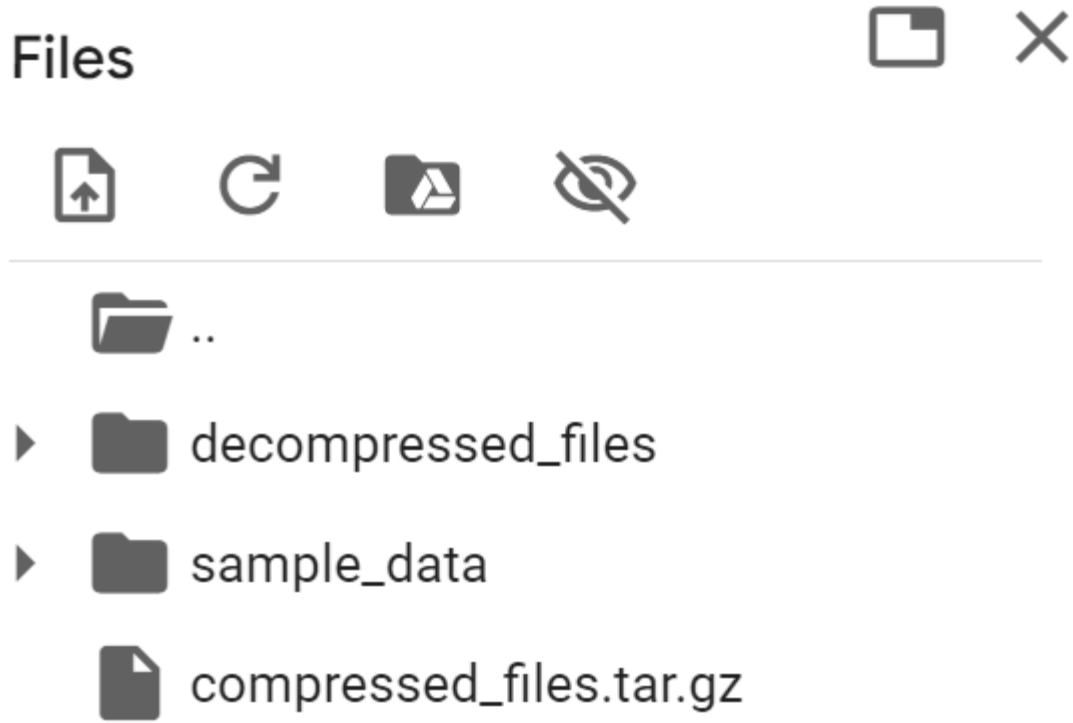
Compressing Files Using tarfile

- The tarfile module is used for creating TAR archives, which can be further compressed using formats like gzip or bzip2.

```
import tarfile

def compress_files_tar(files, output_tar):
    with tarfile.open(output_tar, 'w:gz') as tarf:
        for file in files:
            tarf.add(file)

# Usage
files_to_compress = ['file1.txt', 'file2.txt', 'file3.txt']
output_tar = 'compressed_files.tar.gz'
compress_files_tar(files_to_compress, output_tar)
```


Files
..
decompressed_files
sample_data
compressed_files.tar.gz

Decompressing Files Using tarfile

```
def decompress_tar(input_tar, output_dir):
    with tarfile.open(input_tar, 'r:gz') as tarf:
        tarf.extractall(output_dir)

# Usage
input_tar = 'compressed_files.tar.gz'
output_dir = './decompressed_files'
decompress_tar(input_tar, output_dir)
```

Compressing Files Using Gzip

- The gzip module is used for compressing individual files using the Gzip format.

```
import gzip
import shutil

def compress_file_gzip(file, output_gzip):
    with open(file, 'rb') as f_in:
        with gzip.open(output_gzip, 'wb') as f_out:
            shutil.copyfileobj(f_in, f_out)

# Usage
file_to_compress = 'file1.txt'
output_gzip = 'file1.txt.gz'
compress_file_gzip(file_to_compress, output_gzip)
```

Decompressing Files Using Gzip

```
def decompress_file_gzip(input_gzip, output_file):
    with gzip.open(input_gzip, 'rb') as f_in:
        with open(output_file, 'wb') as f_out:
            shutil.copyfileobj(f_in, f_out)

# Usage
input_gzip = 'file1.txt.gz'
output_file = 'file1.txt'
decompress_file_gzip(input_gzip, output_file)
```

- The shutil.copyfileobj function in Python is used to copy the contents of one file-like object to another.