**Section A: Basic Understanding (with Real-Time & Code Examples)**

**1) What is a Python package? How is it different from a module?**

* A **module** is a single .py file with code you can import.
* A **package** is a **folder** containing one or more modules (files), and an \_\_init\_\_.py file.

Think of a **module** like a single tool (e.g., a screwdriver), and a **package** as a **toolbox** that contains many tools (modules).

**Example:**

**Module (file): calculator.py**

def add(a, b):

return a + b

**Package: math\_utils/ folder**

math\_utils/

\_\_init\_\_.py

basic\_ops.py

advanced\_ops.py

You can import a module:

import calculator

Or a package:

from math\_utils import basic\_ops

**2) What is the purpose of \_\_init\_\_.py in a package directory?**

* It makes the folder recognizable as a Python package.
* It can also be used to initialize package-level variables or control what gets imported.

It’s like a **reception desk** in an office that manages what services (modules) the visitors (imports) can access.

**Example:**

**math\_utils/\_\_init\_\_.py**

print("math\_utils package is loaded")

Now when you import the package:

import math\_utils

You’ll see:

math\_utils package is loaded

**3) What happens when you use from package import \* in Python?**

It imports everything **mentioned in the \_\_all\_\_ list** in \_\_init\_\_.py.

* If \_\_all\_\_ is **not defined**, nothing specific will be imported from submodules.

It’s like saying "Bring me all available tools from the toolbox", but if the toolbox (\_\_init\_\_.py) says “Only give screwdrivers (basic\_ops)”, you’ll only get that.

**Example:**

**\_\_init\_\_.py**

\_\_all\_\_ = ['basic\_ops']

Now:

from math\_utils import \*

basic\_ops.add(1, 2) # Works

advanced\_ops.power(2, 3) # Error: not imported

**4) What is the effect of defining \_\_all\_\_ in a package’s \_\_init\_\_.py file?**

* It **controls what gets imported** when you use from package import \*.

You control what’s shown to guests in your living room. Just because the house has a TV and fridge doesn't mean everyone gets access unless allowed (\_\_all\_\_).

**Example:**

**math\_utils/\_\_init\_\_.py**

\_\_all\_\_ = ['basic\_ops']

Now:from math\_utils import \*

print(add(5, 6)) # works

print(power(2, 3)) # Error

Only basic\_ops is imported via \*.

**5) How can you create and use a subpackage in Python?**

* A **subpackage** is a package inside another package.
* Both the main package and subpackage should have \_\_init\_\_.py.

Imagine a company (shapes) with departments (area) and employees (circle.py, rectangle.py).

**Example Folder Structure:**

shapes/

\_\_init\_\_.py

area/

\_\_init\_\_.py

circle.py

rectangle.py

**circle.py**

def area\_of\_circle(r):

return 3.14 \* r \* r

**Using it:**

from shapes.area.circle import area\_of\_circle

print(area\_of\_circle(5)) # 78.5

**Section B: Coding-Based Questions**

**6) Create a package called math\_utils with the following modules:**

**basic\_ops.py – containing functions for add, subtract**

**advanced\_ops.py – containing functions for power and factorial**

**Demonstrate how to import and use all functions using from math\_utils import \*.**

day11assignment/

├── main.py

├── math\_utils/

│ ├── \_\_init\_\_.py

│ ├── basic\_ops.py

│ └── advanced\_ops.py

├── my\_package/

│ ├── \_\_init\_\_.py

│ ├── module\_a.py

│ └── module\_b.py

├── shapes/

│ ├── \_\_init\_\_.py

│ └── area/

│ ├── \_\_init\_\_.py

│ ├── circle.py

│ └── rectangle.py

math\_utils/basic\_ops.py

# Q6: basic\_ops.py – functions for add and subtract

def add(a, b):

return a + b

def subtract(a, b):

return a – b

math\_utils/advanced\_ops.py

# Q6: advanced\_ops.py – functions for power and factorial

def power(base, exp):

return base \*\* exp

def factorial(n):

result = 1

for i in range(2, n + 1):

result \*= i

return result

**7) Write Python code to show how intra-package references work when module\_a.py imports a function from module\_b.py inside the same package.**

my\_package/module\_b.py

# Q7: Define greet() in module\_b

def greet():

return "Hello from module B!"

my\_package/module\_a.py

# Q7: Import greet() from module\_b using relative import

from .module\_b import greet

def display():

print("Calling from module A →", greet())

**8) Create a package shapes with a subpackage area. In area, create modules circle.py and rectangle.py. Show how to import area\_of\_circle from circle.py in rectangle.py using relative import.**

shapes/area/circle.py

# Q8: Function to calculate area of circle

def area\_of\_circle(radius):

return 3.14 \* radius \* radius

shapes/area/rectangle.py

# Q8: Import area\_of\_circle from circle.py using relative import

from .circle import area\_of\_circle

def display\_circle\_area():

r = 5

print("Area of circle with radius", r, "is", area\_of\_circle(r))

**main.py (Testing All)**

# Q6: Import all from math\_utils

from math\_utils import \*

from math\_utils.advanced\_ops import power, factorial

print(add(2, 3)) # 5

print(subtract(7, 4)) # 3

print(power(2, 3)) # 8

print(factorial(5)) # 120

# Q7: Intra-package call from module\_a → module\_b

from my\_package import module\_a

module\_a.display()

# Q8: Relative import from shapes.area.rectangle → circle

from shapes.area import rectangle

rectangle.display\_circle\_area()

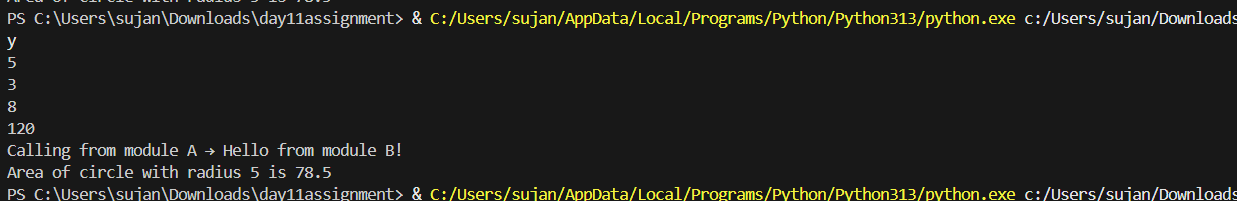
**9)Modify your package math\_utils to include \_\_all\_\_ = ['basic\_ops'] in the \_\_init\_\_.py. What will happen if you run from math\_utils import \* after that?**

math\_utils/\_\_init\_\_.py

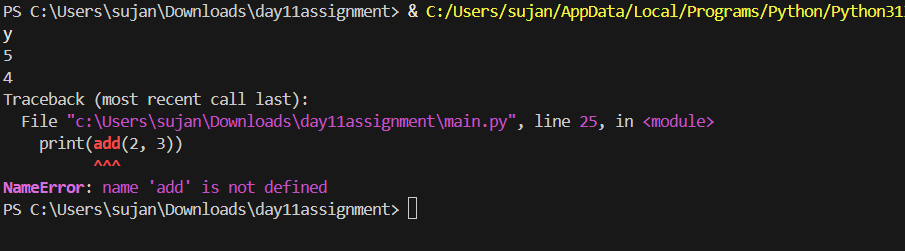
# Q9: Include only basic\_ops in \_\_all\_\_

\_\_all\_\_ = ['basic\_ops']

**OUTPUT**

****

**Output while implementing Q9**

****