Week2 day5

Installed qodo extension -

.toml file -

Patterns -

1.

```
n = 5 # Change this value for different sizes
for i in range(n): # Outer loop for rows
for j in range(i + 1): # Inner loop for columns
print("*", end=" ") # Print star with space
print() # Move to the next line
```

o/p:

```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
```

2.

```
for i in range(n, 0, -1): # Outer loop (rows), starts from n down to 1
for j in range(1, n + 1): # Inner loop (columns)
if j >= i:
print("*", end=" ") # Print star
else:
print(" ", end=" ") # Print space
print() # Move to the next line
```

o/p:

```
      *
     **
   * * *
  * * * *
* * * * *
```

3.

```
n = 5 # You can change this value for different sizes
k = n # Controls the decreasing spaces

for i in range(1, n + 1): # Outer loop (rows)
for j in range(1, n + 1): # Inner loop (columns)
if j >= k:
print("*", end=" ") # Print star
else:
print(" ", end="") # Print space
k -= 1 # Decrease k after each row
print() # Move to the next line
```

o/p:

```
    *
   * *
  * * *
 * * * *
* * * * *
```

4.

```
num = 1

# Outer loop to handle number of rows
for i in range(n):
# Inner loop to handle number of columns
for j in range(i + 1):
print(num, end=" ") # Print number with space
# Increment number at each row
num += 1
# Move to the next line
print()
```

o/p:

1
2 2
3 3 3
4 4 4 4
5 5 5 5 5

`wheel` is a **built-package format** for Python. It helps speed up package installation by providing precompiled packages, avoiding the need to compile source code during installation.

1. Update Package Lists

```
sudo apt update
```

- This refreshes the package list on your system.
- Ensures you get the latest versions of software available from Ubuntu/Debian repositories.

2. Install Python and Virtual Environment Tools

```
sudo apt install python3-venv python3-full
```

- **`python3-venv`**: Provides tools to create virtual environments.
- **`python3-full`**: Installs a complete Python development setup, including standard libraries and dependencies.

3. Remove Any Existing Virtual Environment

```
rm -rf env
```

- Deletes the `env` directory if it exists.
- Ensures you're starting fresh without conflicts from an old environment.

4. Create a New Virtual Environment

```
python3 -m venv env
```

- Creates a virtual environment named `env`.
- A virtual environment is an isolated workspace for Python projects.

5. Activate the Virtual Environment

```
source env/bin/activate
```

- Activates the virtual environment.
- After activation, Python and pip commands will use the isolated environment instead of the system-wide installation.

6. Upgrade Package Tools

```
pip install --upgrade pip setuptools wheel
```

- Updates **pip**, **setuptools**, and **wheel** to their latest versions.
- Ensures you have the latest tools for managing and installing Python packages.

This setup is useful when working on Python projects that require dependency isolation. 🚀