# Running Python Script Continuously in Windows Terminal

## 1. Write a Loop in Your Script:

As mentioned previously, to have your script run indefinitely, the primary step is to wrap your code inside a loop.  
  
```python  
while True:  
 # Your code here  
```  
  
Ensure there's a mechanism to exit the loop, like checking for a specific user input or a keyboard interrupt. You might also want to include sleep intervals using `time.sleep()` to prevent excessive CPU usage.

## 2. Running in Terminal:

Open the Command Prompt or PowerShell and navigate to the directory containing your script. Then execute:  
  
```bash  
python your\_script.py  
```

## 3. Running Script in Background:

Windows doesn't have a direct equivalent to the `nohup` command found in Unix-based systems. However, you can use the `start` command with the `/B` option:  
  
```bash  
start /B python your\_script.py  
```  
  
This will start your Python script in the background. However, if you close the terminal, the script will also be terminated.

## 4. Using Task Scheduler:

Windows Task Scheduler can be used to run your script at specific times or during startup. You can also set it up to restart the script if it fails. While this isn't exactly "continuous," it's a way to ensure that the script keeps running across reboots or if it exits unexpectedly.

## 5. Third-Party Software:

There are third-party tools like `NSSM` (the Non-Sucking Service Manager) that allow you to run scripts (or any executable) as a Windows service. This way, they can start on boot, restart on failure, and run without a logged-in user.

## 6. Monitor Your Script:

Especially if your script is performing crucial tasks, you'll want to set up some monitoring to notify you if it stops running. This might be as simple as logging to a file and checking it periodically, or as advanced as setting up a monitoring solution to alert you of issues.

## 7. Avoid Memory Leaks:

Memory leaks can still occur on Windows, just as on any other OS. Regularly monitor your script's memory consumption and test for potential memory leaks.