این کد یک سرور ساده TCP پیاده‌سازی می‌کند که از مکانیسم بازیابی خطا با استفاده از الگوریتم تحکیم پنجره (Congestion Window) استفاده می‌کند. الگوریتم تحکیم پنجره برای مدیریت تعداد پیام‌هایی که می‌توانند همزمان ارسال شوند و تعیین سرعت رشد این تعداد استفاده می‌شود.

حال خط به خط توضیحات این کد به شرح زیر است:

1. \*\*import socket, threading\*\*: وارد کردن ماژول‌های مورد نیاز برای ایجاد سرور و مدیریت رشته‌ها.

2. \*\*SERVER\_ADDRESS = ('localhost', 12345)\*\*: تعریف یک ثابت به نام `SERVER\_ADDRESS` که آدرس محلی و پورت 12345 را مشخص می‌کند.

3. \*\*CONGESTION\_WINDOW\_SIZE = 4\*\*: تعیین اندازه اولیه پنجره تحکیم (Congestion Window) به عنوان 4.

4. \*\*def handle\_client(client\_socket)\*\*: تعریف یک تابع برای پردازش ارتباط با یک کلاینت.

5. \*\*while True\*\*: یک حلقه بی‌نهایت برای دریافت داده‌های ارسال شده از کلاینت.

6. \*\*data = client\_socket.recv(1024)\*\*: دریافت داده از کلاینت با حداکثر طول 1024 بایت.

7. \*\*if not data: break\*\*: اگر داده‌ای دریافت نشود (ارتباط قطع شود)، حلقه را متوقف کن.

8. \*\*print(f"Received: {data.decode()}")\*\*: نمایش داده دریافتی به صورت رشته.

9. \*\*client\_socket.sendall("ACK".encode())\*\*: ارسال پاسخ به کلاینت برای تأیید دریافت داده.

10. \*\*client\_socket.close()\*\*: بستن اتصال با کلاینت بعد از اتمام ارتباط.

11. \*\*def start\_server()\*\*: تعریف یک تابع برای شروع سرور.

12. \*\*server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)\*\*: ایجاد یک شیء سوکت با تنظیمات مناسب.

13. \*\*server\_socket.bind(SERVER\_ADDRESS)\*\*: متصل کردن سوکت به آدرس و پورت مشخص شده.

14. \*\*server\_socket.listen(5)\*\*: در حالت گوش دادن برای اتصال‌های ورودی.

15. \*\*print(f"Server listening on {SERVER\_ADDRESS}")\*\*: نمایش پیام شروع به گوش دادن به اتصال‌ها.

16. \*\*while True\*\*: حلقه بی‌نهایت برای دریافت اتصال‌های ورودی از کلاینت‌ها.

17. \*\*client\_socket, client\_address = server\_socket.accept()\*\*: دریافت اتصال از کلاینت و آدرس آن.

18. \*\*print(f"Accepted connection from {client\_address}")\*\*: نمایش پیام دریافت اتصال از یک کلاینت.

19. \*\*client\_thread = threading.Thread(target=handle\_client, args=(client\_socket,))\*\*: ایجاد یک رشته برای پردازش کلاینت.

20. \*\*client\_thread.start()\*\*: شروع اجرای رشته.

21. \*\*if \_\_name\_\_ == "\_\_main\_\_": start\_server()\*\*: اجرای سرور اگر این اسکریپت به عنوان اسکریپت اصلی اجرا شود.