

פרויקט: מנתה לוגים של תעבורת רשת

שלב 4 - Generators (yield)

מטרת השלב

עד עכšíו טענו את כל הקובץ לזכרון. זה עובד עם 10,000 שורות, אבל מה קורה עם קובץ של 10 מיליון שורות?

בשלב זה נלמד לעבוד קבצים ענקיים בצורה יעילה באמצעות **Generators**.

הבעיה

הקוד הבא טוען את כל הקובץ לזכרון:

```
def get_all_suspicious(filepath):
    suspicious_lines = []
    with open(filepath, 'r') as file:
        for line in file:
            if is_suspicious(line):
                suspicious_lines.append(line)
    !מחזיר רשימה ענקית # return suspicious_lines
```

אם הקובץ ענק - התוכנית תקרוס או תהיה איטית מאוד.

הפתרון: במקום לאסוף הכל לרשימה ולהחזיר בסוף, נשתמש ב- `yield` כדי להחזיר תוצאות אחת אחר כדי UIBוד.

```
def get_all_suspicious(filepath):
    with open(filepath, 'r') as file:
        for line in file:
            if is_suspicious(line):
                מחזיר שורה אחת וממחכה #
                yield line
```

דרישות שלב 4

בשלב זה נמיר חלק מהפונקציות שכתבנו לשימוש ב-`yield`.

1 קריית קובץ עם `yield`

כתבו פונקציה שקוראת את קובץ הלוג וממחירה (`yield`) כל שורה כרשימה של שודות - בלי לטעון את כל הקובץ ל זיכרון.

2 סינון שורות חדשות עם `yield`

כתבו פונקציה שמקבלת generator של שורות וממחירה (`yield`) רק שורות שיש בהן לפחות חד אחד.

הfonקציה לא טוענת את כל השורות - היא מעבדת שורה, בודקת, אם חשודה - ממחירה אותה.

3 החזרת חדשות עם פרטיו שורה

כתבו פונקציה שמקבלת generator של שורות וממחירה (`yield`) עבור כל שורה חשודה tuple של (שורה, רשיימת_חדשות).

דוגמה לפلت:

```
(  
    ["2024-01-15 03:23:45", "45.33.32.156", "10.0.0.5", "22", "SSH", "6000"],
```

```
[ "EXTERNAL_IP", "SENSITIVE_PORT", "LARGE_PACKET", "NIGHT_ACTIVE" ]
```

```
)
```

4 ספירה בלי טעינה לזיכרון

כתבו פונקציה שמקבלת generator ומחזירה את מספר השורות החשודות - בלי לשמור אותן בזיכרון.

רמז: אפשר לספור עם לולאה פשוטה או עם `sum(1 for ...)`

5 שרשרת generators

חברו את הפונקציות יחד: קריאה → סינון → הוספת פרטימ. הריצו על קובץ הלוג ומספרו כמה שורות חשודות יש.

דוגמה לשימוש:

```
lines = read_log("network_traffic.log")      # generator
suspicious = filter_suspicious(lines)        # generator
detailed = add_suspicion_details(suspicious) # generator

count = count_items(detailed)
print(f"Total suspicious: {count}")
```

למה זה חשוב? במודיעין עובדים עם קבצי לוג ענקיים - ג'אגה בייטים ואפילו טריהבייטים. אי אפשר לטעון הכל לזיכרון. generators מאפשרים לעבוד כמוניות אינסופיות של מידע בצורה יעילה.