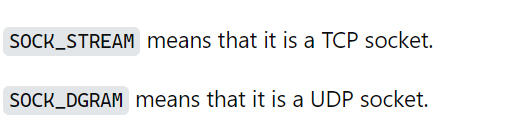
התחלנו עם הserver :

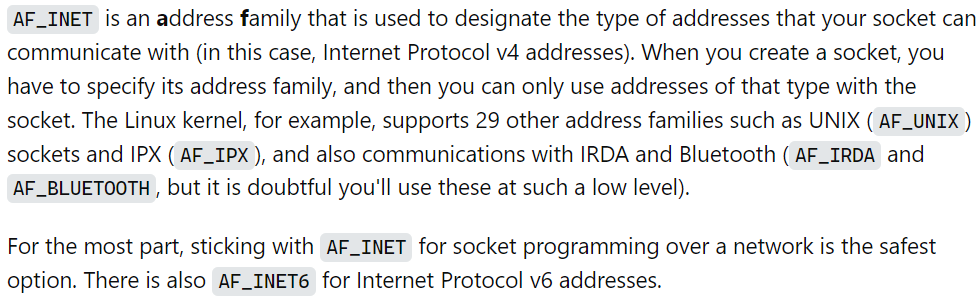




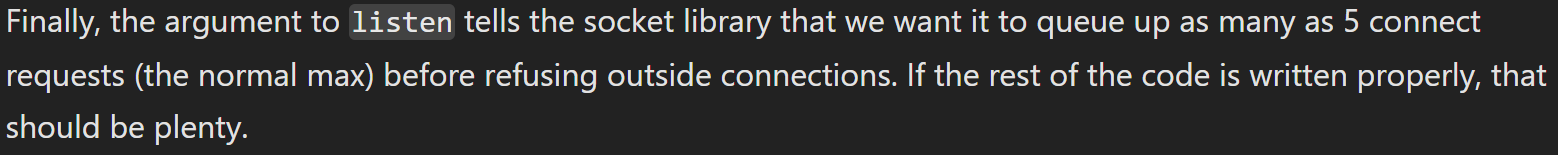


פקודה לפתיחת socket חדש כאשר:

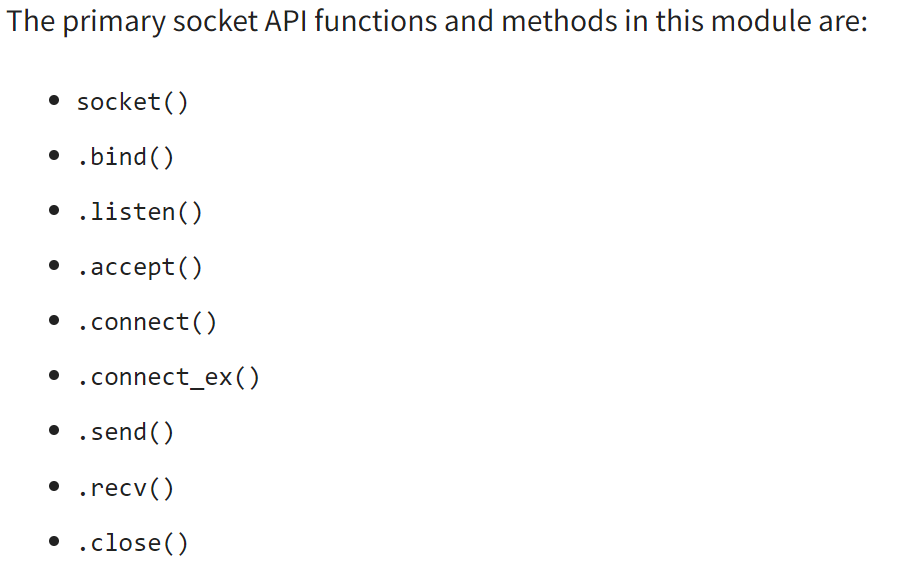




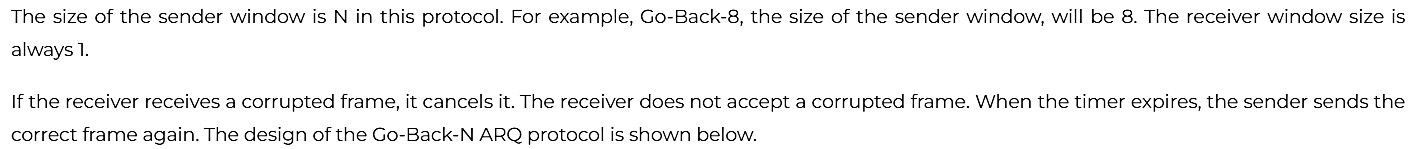






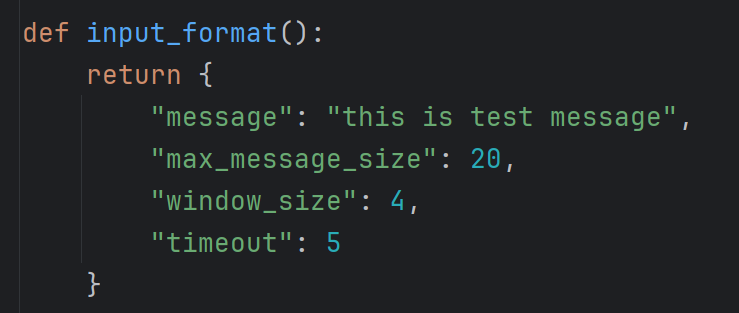


במחקלת socket:

Sliding window protocol – go-back-N ARQ:

תמונה שמכילה טקסט, תרשים, צילום מסך, קו

התיאור נוצר באופן אוטומטי

קביעת הפורמט :



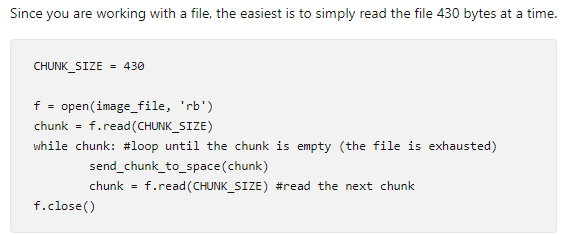
אחרי שיצרתי חיבור שלחנו אותו לשרת דרך הסוקט, יש צורך בלקודד אותו על מנת להעביר אותו.

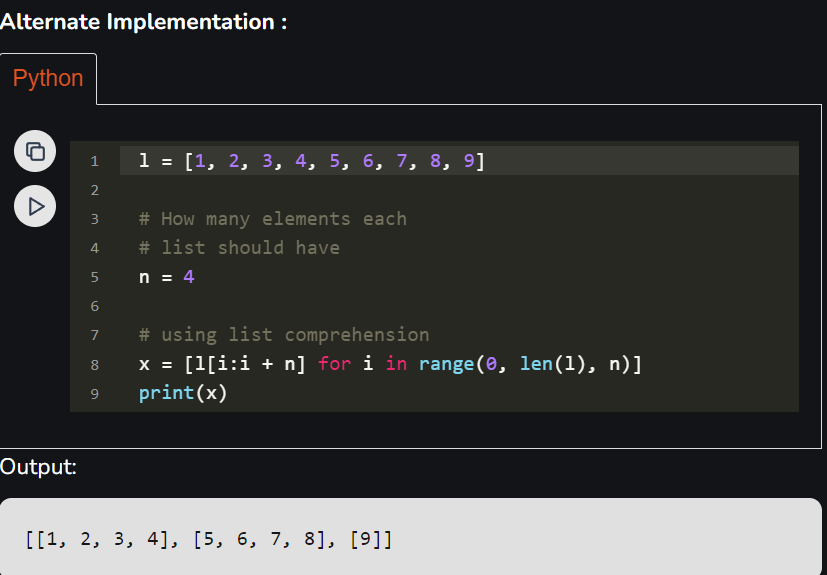
בצד שרת:

ווידוא שהגיע

עכשיו צריך להבין איך לחלק את ההודעה לצ'אנקים לפי כמו הבתים שנקבעו.

בקובץ:





במשתנים:

A screenshot of a computer

Description automatically generated

על מנת שיהיה קל לקרוא מידע מהקובץ השתמשנו בjson שימיר את המידע לdictionary

* מה שאומר – שלפי בקשת המטלה, נצטרך שבקובץ כל מידע יצטרך להיות בשורה בקובץ.
* מגבלה נוספת שיש, על מנת לוודא בשרת שהוא מקבל את המידע בסדר הנכון אז קודדנו את המידע בצורה כזו:
  + {seq\_number}|"massage"
* מה שאומר שאם ההודעה ממש ארוכה, נצטרך שגודל הבתים המקסימלי לכל הודעה צריך להיות מספיק גדול כדי שיהיה מקום גם להודעה וגם למספר הודעה.
  + לדוגמה: 245|"poop"
  + צריך לוודא שההודעה כולה עומדת בתנאי של שליחת גודל הקבצים המקסימלי. שלא יווצר מצב שבו יש מקום רק למספר ההודעה בלי להוסיף עוד מידע על זה.

ארכיטקטורה –

* נפעיל שרת, בשרת נחכה לחיבור של לקוח. ברגע שקיים חיבור הלקוח בוחר את סוג תצורת שליחת ההודעה:
  + 1 – input by keyboard
  + 2 – input by files
* במקרה והלקוח בחר 1 – השרת יקליד את גודל ההודעה המקסימלי וישלח ללקוח.
  + הלקוח יקבל ויכבד את בקשת השרת לגודל המקסימלי.
  + נבחר את הקלט שנרצה להעביר לשרת, גודל החלון, ומה זמן שהוא צריך לחכות על מנת לנג'ס שוב לשרת.
* במקרה והלקוח בחר 2 – השרת הלקוח יקראו את המידע הרצוי מהקובץ שנגיד לו לפי הpath שלו.
  + בשלב הזה הלקוח מפענח את המידע שיש בקובץ וממיר למשתנים ומעביר את המידע הזה לפונקציה של החלון.
  + השרת עושה את זה באופן דומה אך וק לגודל הבתים המקבמלים שנשלח.

Client.py –

* 1 – input from keyboard
* 2 – reading from file : fopen to read the file, and json for easier translation to variables.

Server.py –

* Sames as client for 1 and 2

Wireshark readings:

A screenshot of a computer

Description automatically generatedserver:

A black background with a line of lights

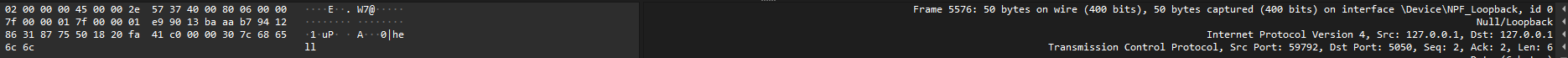
Description automatically generatedclient:

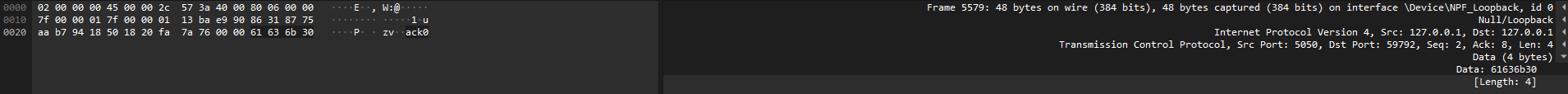
A screen shot of a computer

Description automatically generated

A screen shot of a computer

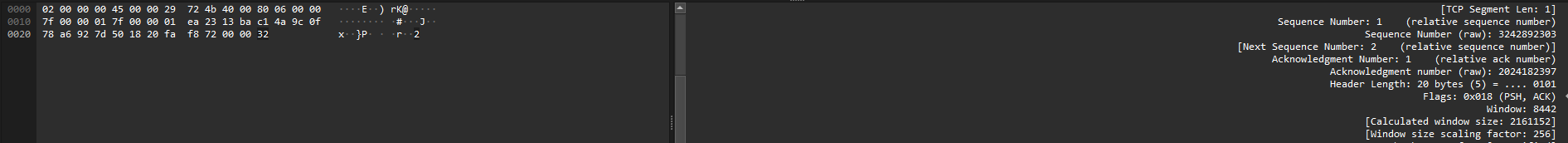
Description automatically generated

הלקוח שולח הודעה 0 לשרת. ואכן גודל כל הודעה הוא 6 בתים לבקת השרת.

שליחת הודעה הראשונה מהלקוח לשרת

קבלת הודעה 0 מהשרת.

עבור קריאה מקובץ:



A screenshot of a computer

Description automatically generatedהשרת מקבל 2

נשלחת הודעה לשרת בגודל 20 בתים לפי הנדרש. וקבלת ההודעה מהשרת עם ack.

A black rectangular object with white lines

Description automatically generated

* כמובן ההקלטות יתווספו לקובץ זיפ שתוכל להתרשם מהם בעצמך.

A black and blue line

Description automatically generated with medium confidence

לאחר סיום התקשורת השרת מזהה שהיה סגירת חיבור.

A screen shot of a computer

Description automatically generated

דוגמה למקרה בו בכוונה שיבשנו את סדר פעולת השליחה של המידע, על מנת לראות שבאמת שהלקוח אכן מזהה שלא קיבל את מספר הack הנכון, ושולח שוב לשרת את מה שעוד לא קיבל בחלון מחדש.

מקורות:

* <https://stackoverflow.com/questions/35725732/what-is-the-function-of-sock-stream>
* <https://docs.python.org/3/howto/sockets.html>
* [https://realpython.com/python-sockets/](https://realpython.com/python-sockets/?utm_source=chatgpt.com)
* <https://www.javatpoint.com/sliding-window-protocol>
* <https://www.reddit.com/r/learnpython/comments/51hxul/how_do_you_split_a_binary_file_into_specific/>
* <https://www.geeksforgeeks.org/break-list-chunks-size-n-python/>
* <https://www.w3schools.com/python/python_json.asp>