**מבוא ל-python**

**,**

שפת Python מוגדרת כשפת interpreter (פרשן) ומתרגמת את השורות\פקודות אחת אחרי השנייה - משמעות הדבר שבמקרה של טעות הפקודות עד מקום הטעות יבוצעו והמשך הקוד יקרוס.

דרישות מערכת:

Python לא מגיע כברירת מחדל ב-Windows ויש צורך בהתקנתה.

לאחר ההתקנה לצורך נוחות ניתן ואף מומלץ להוריד סביבת עבוד.

סביבת העבודה המומלצת ביותר ע"י המרצה הינה PyCharm עבור מערכת הפעלה ווינדוס.

לצורך נוחות ישנה הגדרה ספציפית אשר מגדירים ב-PyCharm הנמצאת תחת הנתיב הבא:

File-> Settings-> Editor-> General-> mark checkbox zoom font.

תחילת עבודה:

1. שלב ראשון - יצירת פרויקט.
2. שם פרויקט ואישור.
3. Right-click on project-> New-> Python File
4. לתת שם לקובץ.

הפקודה print הצגת פלט של מידע.

---------------------------

תרגיל: כתבו תוכנית אשר מדפיסה My Name Is Ben.

---------------------------

Python -ברגע שהריץ פקודה אינו זוכר את הפקודות הקודמות.

בכדי להתגבר על הלוגיקה הנ"ל בשפות תכנות מוגדרים משתנים (variables)/

משתנה - הוא מקום בזיכרון אשר מאחסן מידע לשימוש מאוחר יותר.

לכל המשתנים ב-Python ישנם שלוש הגדרות:

1. שם.
2. ערך.
3. סוג.

הגדרת משתנה ב-Python מוגדרת ע"י הסימן '=' שמשמעותו השמה, תחת הנוסחה הבאה:

Var = Value.

Python - מגדיר את סוג המשתנה באופן אוטומטי.

סוגי משתנים:

כפי שציינו לכל משתנה מוגדר סוג לפי הכללים הבאים:

Int - מדובר על כל המספרים. (Python מגדיר את סוג המשתנה לפי הערך).

Float- כל המספרים העשרוניים.

Long- מספרים ממש ממש גדולים.

String- כל המחרוזות. מהי מחרוזת- Text. מוגדר ע"י גרשיים.

Boolean- אמת או שקר.

בהינתן הודעת שגיאה לקרוא את תוכן ההודעה.

ישנה אפשרות ה-Python לבצע הקצאה מרובה לדוגמה a,b,c = 6

הפקודה print type(a) מקבלת משתנה ומחזירה מה הסוג שלו.

מניפולציות של משתנים:

בניגוד למספרים העבודה עם מחרוזות מבחינת שפת התכנות היא רחבה יותר.

מה ניתן לבצע עם מחרוזות:

1. שרשור (חיבור שתי מחרוזות). לדוגמה- str1 + str2
2. שרשור כפול (ידפיס כפול מספר הפעמים). "hi" \*3 יהיה "hihihi".
3. חיתוך\פילוח. ב-Python ישנה יכולת להתייחס לחלק מ-String ולא לכולו ההתייחסות מבוצעת ע"י סוגריים מרובעים וציון אינדקס. מהו אינדקס? מיקום יחסי החל מאפס. לדוגמה str1[0:6:2} יכלול מאפס עד שש לא כולל שש בקפיצות של 2.

או str1[ 4: ] מהתו הרביעי עד הסוף או str1[ :4 ] מההתחלה עד התו הרביעי או str[ ::2 ] מההתחלה עד הסוף בקפיצות של 2.

* אינדקס יכול להיות שלילי ב-Python והמשמעות מהסוף להתחלה לדוגמה str [-3]מסמל את התו השלישי מהסוף.

**Casting - המרה:**

ב-Python יש את היכולת לבצע המרה בין משתנים, הדבר מתבצע באופן הבא:

num="6"

int (num)

wow = 32

str(wow)

Num = “72”

float(int(nu))

מציאת טקסט:

ל-Python יש את היכולת לחפש String בתוך String הדבר מבוצע ע"י הפקודה find.

ישנם שני מקרי קצה, אחד המילה הרצויה מופיעה כמה פעמים במקרה הזה ה-index בהופעה הראשונה.

שני המילה כלל לא מופיע.

---------------------------------------

שיעורי בית:

1. כתבו טקסט מאוד ארוך והפעילו עליו את כל אחת מן המניפולציות שלמדנו בשיעור.

2. התקינו Python ו-PyCharm.

3. כתבו python challenge בגוגל - פיתרו את 2 האתגרים הראשונים.

---------------------------------------

Split - היא פונקציה אשר מפצלת במקום ספציפי את ה-String וממירה ל-list.

לדוגמה

* ישנו סוג משתנה מורכב בשם מילון (Dictionary) אשר מוגדר באופן הבא:

dict1= {"key":"value"}.

ניתן באמצעות print בהדפסה של string לבצע שילובי משתנים באמצעות key and value

המפתח מתבטא ב-

1. %s
2. %n
3. %d

כאשר s% יוחלף במקומו String.

d% יוחלף במקומו מספר.  
לאחר ה-String יש להגדיר- (value) + %.

לדוגמה:

(print "Hi My Name Is %s" %(Aviad

Replace - הינה פונקציה אשר מקבלת 2 ערכים הערך הראשון מסמל "מה להחליף" והערך השני "במה להחליף".

("Str1.replace("value1", "value2

עבודה עם מספרים:

אופרטורים- כאשר מדברים על הפעולות אשר ניתן להחיל על מספרים ניתן לסכם שפעולות אלו הן מתמטיות בלבד ועל כן אופרטורים הינם סימנים שיש להם משמעות מתמטית .

האופרטורים מתחלקים לקבוצות הבאות:

1. אופרטורים אריתמטיים - כל אחד מפעולות החשבון המוכרות , + חיבור plus ,  
    - חיסור minus ,  
    \* כפל multiply ,  
    / חילוק division ,  
    \*\* חזקה power ,

% modulu .

חילוק - ב-Python גרסה 2.7 החילוק הוא כמו רוב שפות התכנות, משמע החילוק הוא בשלמים 2 = 15/7, במידה ורוצים שהתוצאה תהיה מדויקת אחד מהאיברים צריך להיות float

בנוסף לחילוק, ישנו אופרטור "%" שמשמעותו שארית החלוקה 15%7 יניב את התוצאה 1.

2. אופרטורים לוגיים - הינם סימנים שהתשובה שלהם היא אמת או שקר לדוגמה num > num2 יש . רק 2 תשובות אפשריות נכון ולא נכון.

3. אופרטורים בינאריים -אופרטורים אלו הינם פעולות שהתוצאה שלהם קשורה לבינארי.

וגם & &&

או ||

XOR ^

NOT !

רשימה list:

רשימה הינה משתנה אשר מכיל בתוכו מספר משתנים, אין הגבלה פורמלית לכמות המשתנים ואין הגבלה לסוגי המשתנים.

גישה לפריטים ברשימה ניתן לבצע ע"י אינדקס 0, 3, 6.

ניתן לגשת למספר מקומות באופן הבא:

[lst = [0,15,17,56

[print lst[0:2

ידפיס את 2 הערכים הראשונים.

( print len(lst

ידפיס את מספר האיברים ברשימה.

(Lst.append(variable/value

יוסיף איבר לרשימה מצד ימין.

(print lst.reverse(value

יהפוך את סדר האיברים ברשימה.

lst.pop

יחסיר\יוציא את האיבר האחרון מהרשימה.

(lst.insert(place,what

דוגמה- ("lst.insert(2,"hi

דוחף משתנה במקום נתון (אינדקס).

[del lst[index

מוחק ערך מתוך הרשימה.

--------------------

תרגילים:

1. צור קוד אשר מקבל מספרים לרשימה וחשבו את המידע הבא (4 איברים) להדפיס ממוצע סכום ואורך.

2. הגדירו רשימה והדפיסו אותה הפוך לאחר מכן הוסיפו 10 איברים הדפיסו את את הראשון והאחרון והסירו את החמישי והשישי.

-------------------

**Tuple:**

הוא סוג משתנה כמו list אבל immutable (לא ניתן לשינוי).

שימושי, כי הוא דרך נוחה לוודא שמספר משתנים לא משתנים בהמשך הקוד, מגן על אמינות הנתונים וגם עניין של יעילות.

מוגדר באמצעות סוגריים עגולים.

ניתן להדפיס את כל ה-Tuple וניתן להתייחס לחלק ממנו.

שבירת ה-Tuple :

ניתן לפרק את ה-Tuple למספר משתנים באופן נוח.

Tup = (1,2,3)

a,b,c=tup

**מילון Dictionary:**

כהגדרתו הינו משתנה המכיל מספר איברים, כל איבר מוגדר ממפתח key וערך value

מילון מוגדר באמצעות סוגריים מסולסלים.

{dict1 = {"name":"Aviad","age":30

פונקציות רלוונטיות למילון:

בכדי להדפיס ערך רלוונטי למפתח:

["Print dict["name

מפתח חייב להיות ייחודי.

ניתן להגדיר מילון גם בצורה הבאה:

(dict2 = dict(name="Aviad",Age=30

ישנה פונקציה אשר בודקת האם מפתח מסוים קיים:

("dict1.has.key("id

בכדי להדפיס את כל המפתחות יש להשתמש בפונקציה הבאה:

()print dict1.keys

פונקציה אשר מחזירה את כל האיברים key + value

()print dict1.items

פונקציה אשר מחזירה את כל ערכי המילון:

()print dict1.values

בכדי לקבל ערך של מפתח מסוים:

["print dict1 ["name

("print dict1.get("name

בכדי למחוק איבר ספציפי:

["del dict1["key name

ניתן לנקות את כל המילון:

()dict1.clear

מציג את כל הפונקציות האפשריות עבור המשתנה:

(print dir(dict1

הצגת הסבר על הפונקציה:

(help(dict1.values

**קלט מהמשתמש:**

עד לעת הזו כל התוכניות שכתבנו הינם תכניות מוגדרות להם מידע קודם.

בכדי ליצור תוכנית כללית יותר אשר תתאים למספר אנשים ולא לאחד, יש צורך להכיר את המילה קלט.

ניתן לקלוט ב-Python 2.7 קלט מהמשתמש באופן הבא:

1. במידה ורוצים לקבל מספר הדבר יתבצע ע"י הפקודה "input"

()num = input

print num

2. במידה ורוצים לקבל מחרוזת (String) הדבר יתבצע ע"י הפקודה:

()str = raw\_input

print "Hello", name

בכדי להפוך את התוכנית לידידותית למשתמש יש צורך להדפיס למשתמש הוראות, גם לדבר זה יש 2 אופציות:

1. להדפיס הוראה:

()name = raw\_input

print "What is your age?", name

2. להכניס טקסט בתוך הסוגריים:

name = raw\_input("Hello Aviad")

פונקציית input הינה פונקציה שמוגדרת כ-blocking.

פונקציות Blocking הינם פונקציות שמונעות את המשך הקוד עד שיושלמו.

-------------------

ש.ב:

1. כל התרגילים שביצענו בכיתה לבצע אותם עם קלט מהמשתמש.

2. צרו לעצמכם תרגיל שיחייב שימוש ב3 אופציות לפחות או של list או של tuple או של dictionary

צרו תוכנית אשר מקבלת ערכים למילון, הדפיסו את כל הערכים הדפיסו את כל האיברים.

3. סכמו במילים שלכם והסבירו לעצמכם את שנלמד בשיעורים.

------------------

**תנאים:**

להשלים חוסר הגעה מאוחרת 18:30

קשרים לוגיים וטבלת אמת:

טבלת אמת הינה טבלה אשר מפשטת קשרים לוגיים בין תנאים.

XOR:

True + true = false

False + false = false

True + False= true

False + true = true

NOT:

Not false = true

Not true = false

**לולאות loop:**

כחלק מיכולת שפת התכנות הומצא הרעיון של לולאות

לולאות - קטע קוד המסוגל לרוץ מספר פעמים.

1. For

2. While

לולאת while היא לולאה שרצה כל עוד תנאי מתקיים.

תרגיל - צרו תוכנית אשר סופרת כמה פעמים המשתמש הכניס את האות a.

:<while <condition is

do staff

לולאת for משומשת כאשר המשתמש מעוניין לבצע פעולה מספר מוגדר של פעמים.

לולאת for מאפשרת למתכנת להתייחס לכל משתנה ברשימה ולבצע עליו מניפולציות.

for var in aviad

print var

will print aviad

פקודת break שוברת לולאות.

------------------

ש.ב להגשה:

----------------

**קריאה מקבצים וכתיבה לקבצים:**

Python יכול לבצע מניפולציות על קבצים.

פונקציית open מקבל מיקום ומצב לפתיחה

מצבי open אפשריים:

1. "r" - קריאה מתוך הקובץ.

2. "w" - כתיבה לתוך הקובץ.

3. "a" - להוסיף תוכן לתוך הקובץ בסוף.

4. "rb"- read bytes.

5. "wb" - write bytes.

location = r"C:\Users\Aviad\aviad.txt"

file.read() #return the content of the file

(file.write(this is very long

תרגילים:

1. מה סוג האובייקט שמוחזר ע"י הפונקציה open , רמז השתמשו בפונקציה type .

2. צור script שמעתיק קובץ לקובץ אחר.

3. צור סקריפט שקורא קובץ של מילים לשיר ומדפיס מילה מילה.

4. כתוב סקריפט המקבל מיקום של תמונה מהמשתמש ומייצר העתק של אותה תמונה.

5. בדקו בגוגל על הפונקציה הבאה:

(with open (location,mode

for line in file

print line

6. קבלו תיקיה מהמשתמש המכילה 4 קבצים בדקו האם קובץ אחד הוא העתק של קובץ שני.

**ספריית os:**

ספרייה הינה הגדרה מדויקת לייבא קוד שמישהו אחר כתב כאילו הוא נכתב מקומית.

בכדי לייבא את הספרייה משתמשים בפקודה import.

בכדי לגשת לקוד בספרייה יש להשתמש בשם הספרייה ולאחריו לרשום נקודה "os."

ספריית OS מכילה כמות אדירה של פונקציות.

(os.rename(path1,path2 מקבלת קובץ ומיקומו ומשנה את השם שלו.

()os.walk

()os.chdir משנה את התיקייה שמושפעת ע"י הקוד.

()os.mkdir מייצר תיקייה.

()os.rmdir מוחק תיקייה.

()os.getcwd מחזיר את התיקייה המושפעת כרגע.

()os.path.curdir מחזיר את התיקייה המושפעת כרגע.

()os.path.isdir מקבל נתיב ומחזיר אם זה תיקיה True ובמידה והנתיב הוא לא תיקייה False.

()os.path.isfile מקבל נתיב ומחזיר אם זה קובץ True ובמידה והנתיב הוא לא קובץ False.

()os.path.exists האם הנתיב קיים.

()os.path.getsize מחזיר את גודל הקובץ.

**פונקציות Functions:**

פונקציות הינם השם המקצועי לקטע קוד שמוכר תחת שם אחד.

מהם פונקציות - הצהרה על עבד(הפשטה של המרצה - לא הגדרה מקצועית).

:()<Def <function\_name

#what to do

:()Def print\_hello

("ui = input(“please enter a number

:(for i in range (ui

"print “hello

()print\_hello

Scope:

הינה ההגדרה המקצועית לשאלה היכן יוכר המשתנה.

משמעות: מי בקוד מכיר אותי.

כלל - ככלל כל משתנה אשר מוגדר ב-Main ה-Scope שלו הוא כל הקוד.

ישנה דרך לגרום למשתנה של פונקציה להיות מוכר לכל הקוד- יש לרשום לפני המשתנה global.

:()def print\_hello

global ui

("ui = input(“please enter a number

ישנה בעייתיות בהגדרה Global:

1. בהצהרת משתנה גלובלי ישנו סיכון בדריסת משתנים בעלי אותו שם, לשם כך יש פתרון הפתרון נקרא "return".

2. במקרה של כפילויות שמות משתנים המשתנה המקומי(בתוך הפונקציה) תמיד יגבר על המשתנה הכללי.

פונקציה בנוסף יכולה לקבל משתנים מבחוץ.

-------------------

תרגיל:

כתוב פונקציה בשם פלוס המקבלת 2 מספרים מבחוץ ומחזירה את החיבור ביניהם.

כתוב פונקציה בשם calc המקבלת 2 מספרים וסימן מבחוץ ומחילה את הסימן על 2 המספרים ומחזירה את את התוצאה.

כתוב פונקציה בשם factorial המחזירה את הערך של 5 עצרת למספר שיתקבל מהמשתמש.

------------------

**:()os.walk**

ממיין נתיב לאחר סריקה.

סורק את הקבצים ומבין האם קובץ הוא קובץ או תיקייה.

נכנס לתוך התיקיה.

**Index יצירת לוג לקבצים:**

בכדי לא לחזור על אותן פעולות ובכדי למנוע סריקה מיותרת הומצא ה-Concept של Index (רישום טקסטואלי של הנתיבים הקבצים והתיקיות במערכת).

----------------

תרגיל לבית - סרקו את המערכת וצרו קובץ טקסט עם כלל הממצאים במידה והקובץ טקסט קיים אשרו למשתמש צפייה בתוכנו, בנוסף שאלו את המשתמש האם לעדכן את קובץ הטקסט.

----------------

**Socket**

הוא מונח לוגי המייצר צינור מידע בין שרת לוקח

Socket.socket

משתנה ראשון ייצג סוג חיבור

TCP/UDP

המשתנה השני שלי ייצג איזו כתובת IP הוא מחובר

IPV4/IPV6

רוב החיבורים יהיו TCP ל-IPV4 לכן כותבי הסםריה נתנו את האופציה לתת סוגריים ריקים.

.connect((<tuple>,IP,port #))

Path for connection, is happening with this command

.send(“message”)

שולח הודעה מסויימת

.recv(0-65535)

כמות קיבולת מידע (חייב שיהיה מוגדר מראש).

--------------------------------------------------

תרגיל:

IP: 10.0.4.49

Port: 4443

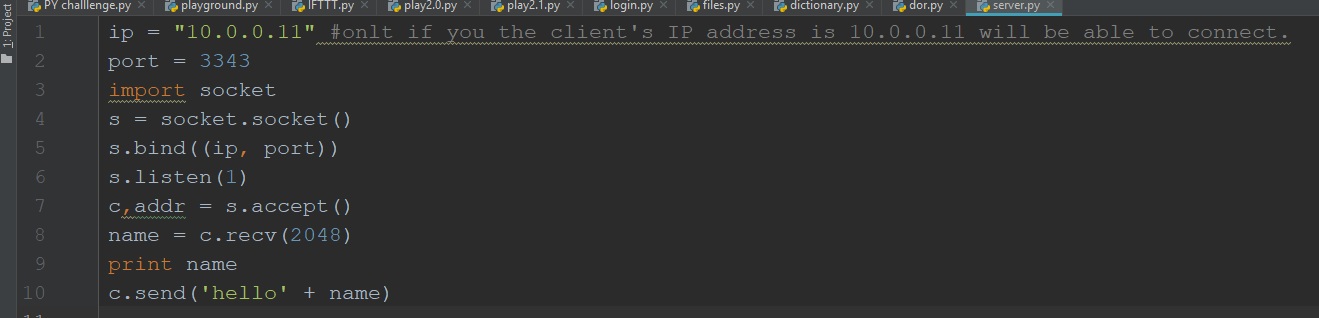
התחברו למחשב שלי

שלחו את השם שלכם והדפיסו את מה שאתם מקבלים על השרת.

------------------------------------------------------------

**יצירת שרת באמצעות Socket**

במקרה הזה: S הוא משתנה של- ()socket.socket



ישנן כתובות IP מיוחדות ומייצגות חיבור מסוים:

א) 127.0.0.1 היא כתובת של הכרטיס רשת שלי עצמו.

ב) 0.0.0.0.0 - כל מי שרוצה להתחבר לשרת שלי יכול להתחבר.

#############################

ה-S מייצג לי חיבור כללי של השרת שכל הלקוחות מתחברים דרכו, מכיון שזה יותר מידי כללי במידה והשרת רוצה לדבר עם לקוח ספציפי python יצרו את

s.accept

var = s.accept()

מאשר חיבור ומקבל TUPLE המכיל 2 אלמנטים

1. מקבל את הSOCKET של הלקוח
2. מקבל את ה-IP של הלקוח (ועוד פרטים כמו פורטים וכו')

ולכן קבלת המידע תתבצע בצורה הבאה:

c.recv(2048)

c.send (‘hello Client’)

c.close()

------------------------------------------

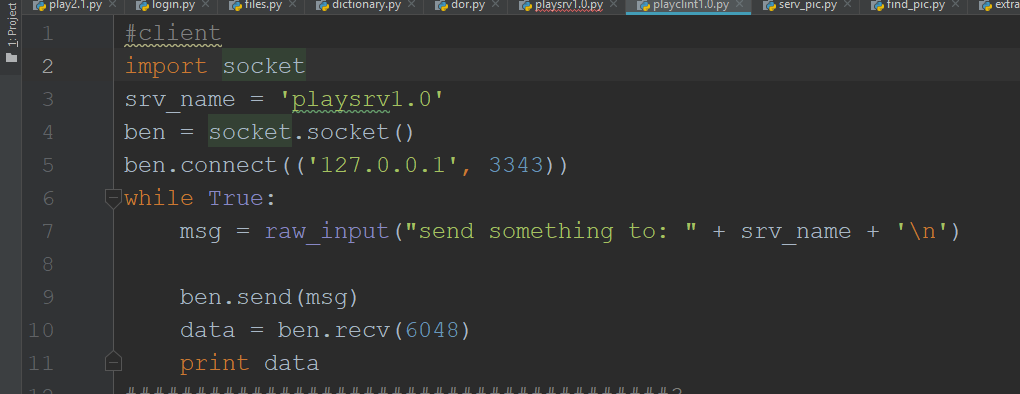
תרגילים להגשה:

**\*לכל אחד מהתרגילים צרו שרת ולקוח.\***

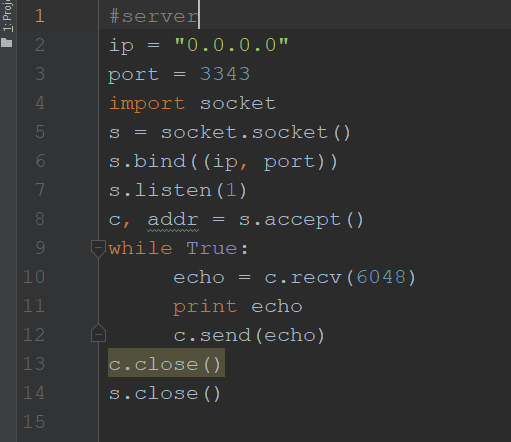
1. צרו שרת ECHO, מה שהלקוח שולח חוזר אליו
2. צרו שרת אשר מקבל מהלקוח פקודה מריץ אותה על המחשב ומחזיר ללקוח את התשובה.
3. צרו שרת אשר מקבל מהלקוח שם של תמונה, מחפש את התמונה, ושולח ללקוח את התמונה, במידה והתמונה לא קיימת החזירו: לא קיים.
4. צור שרת אשר מקבל שם משתמש וסיסמה. במידה והוא צודק הדפיסו לו ברכה אישית, במידה ולא הדפיסו לו שגיאה.
5. סרקו את המערכת וצרו קובץ טקסט עם כלל הממצאים במידה והקובץ טקסט קיים אשרו למשתמש צפייה בתוכנו, בנוסף שאלו את המשתמש האם לעדכן את קובץ הטקסט.

*תשובות לתרגילים:*

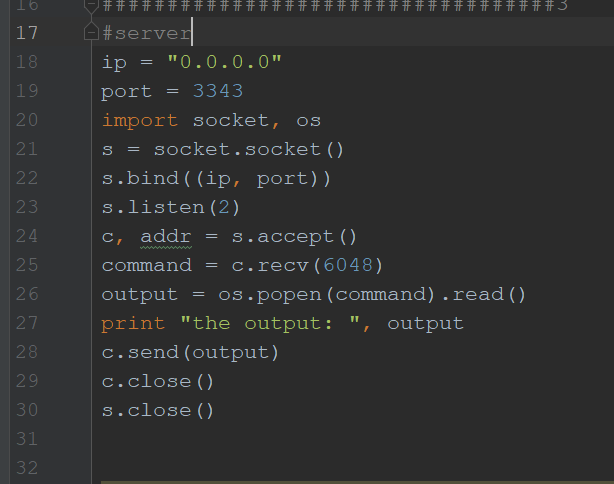
1. לקוח:



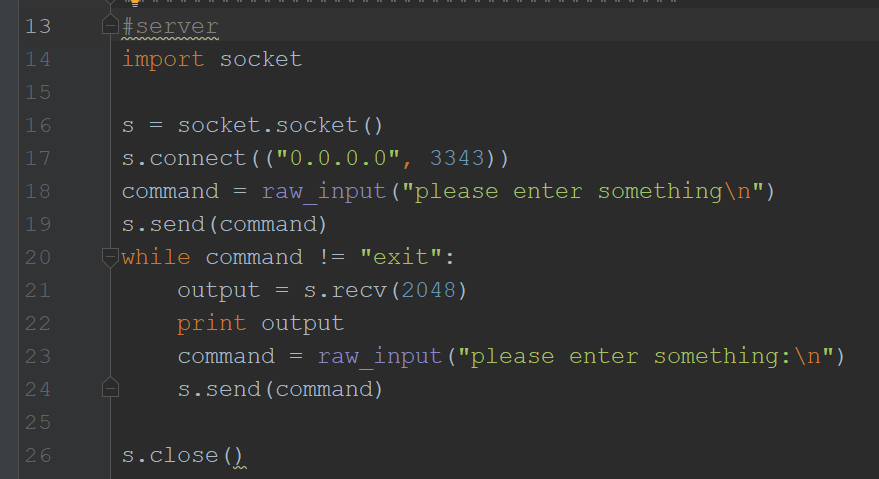
שרת:

.

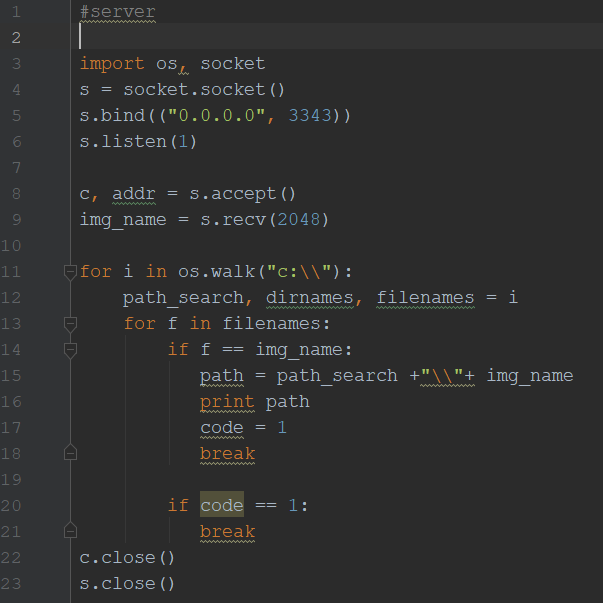
2) שרת:



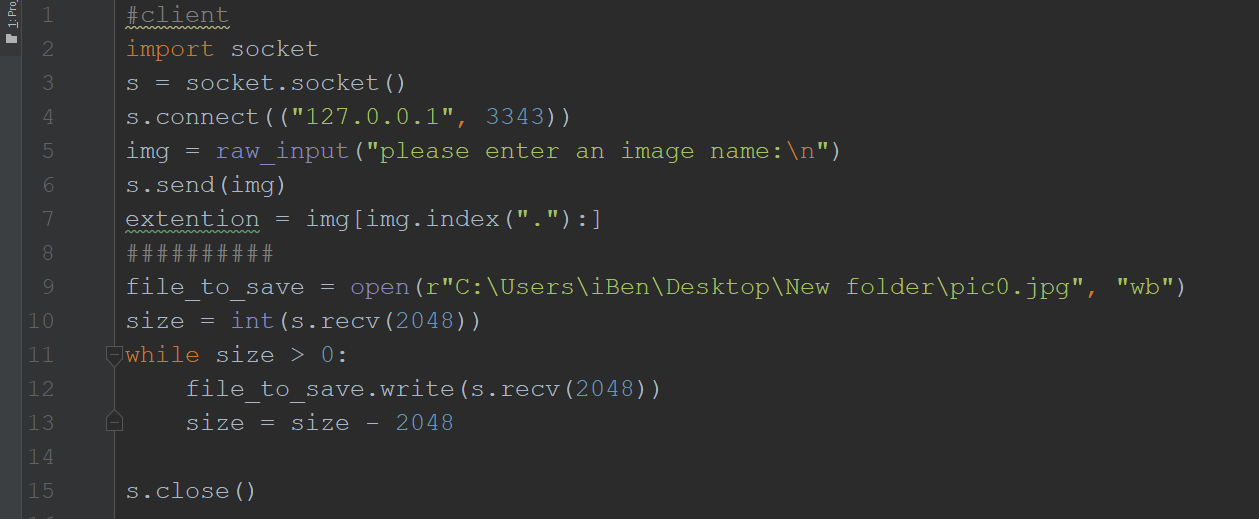
לקוח:

.

3) שרת:



לקוח:

.