**פייתון - הרצאה מספר 6**

**פונקציות**

**הגדרת פונקציה:**

פונקציה/ פרוצדורה - הינה רצף הוראות המבצעות תת משימה מסוימת וניתנת לזימון חוזר בהתאם לתהליך פתרון בעיה.

**הגדרת מתודות- לשם מה?**

**חסכון בכתיבה** - כאשר יש צורך לבצע אותה תת משימה מס' פעמים נגדיר אותה פעם אחת כמתודה ונשתמש בה בכל מקום נדרש. כך נעשה שימוש חוזר במתודה במקום לכתוב מחדש את אותו קטע קוד.

**כתיבה מובנית** (מודולרית) – באמצעותה ניתן לחלק בעיה גדולה לתת-משימות. כל תת משימה תטפל בבעיה מוגדרת. כך גם קל יותר לתקן שגיאות, היות ולכל פעולה תפקיד משלה ומשתנים מקומיים השייכים רק לה. כמו כן, הדבר מאפשר בדיקת נכונות של מודול קטן בכל שלב פיתוח.

**מבנה כללי של פונקציה:**

**def** שם הפונקציה ( ):

**""" docstring - documentation string"""**

גוף הפונקציה - ההוראות לביצוע

**הסבר:**

מתחילים עם המילה השמורה .defלאחר מכן שם הפונקציה (הקפידו לתת שם משמעותי) ואח"כ סוגריים שבתוכם אפשר לשים פרמטרים.

docstring – documentation string - בתחילת הפונקציה יש לכתוב תיעוד, שמתאר מה הפונקציה עושה. כותבים documentation string בין שלושה סימני גרשיים כפולים רצופים, כמו בדוגמה. כדי שאם נעשה help לגבי שם הפונקציה, נקבל בחזרה את ה- docstring שלה. זה מאד מועיל כאשר אנחנו רוצים לשתף קוד.

**לדוגמה:** עבור הגדרת פונקציה זו

def mul(a,b):

**"""Multiplies The Args a,b """**

return a\*b;

**#help(thing) – prints help about that topic**

help(mul)

יוצג הטקסט הבא:

Help on function mul in module \_\_main\_\_:

mul(a, b(

Multiplies The Args a,b

**זימון פונקציות:**

שם הפונקציה (פרמטר / פרמטרים אם ישנם ) **פורמט:**

לזימון פונקציה, יש לרשום את שמה ואחריה סוגריים. אם יש פרמטרים מועברים, יש לציין אותם ואין אין הסוגריים יהיו ריקים.

* פונקציה יכולה להיות מוגדרת עם פרמטרים או בלי פרמטרים.

דוגמה לפונקציה בלי פרמטרים:

def hello():

**"""print hello universe!"""**

print("hello universe!")

hello()

**יודפס:** hello universe!

* דוגמה לפונקציה עם פרמטרים:

def print\_full\_name(fname,lname):

"""print full name

Args:

fname - first name

lname - last name

"""

print(fname+lname)

# בזימון הפונקציה יש לשלוח את אותו מספר פרמטרים כפי שהוגדר בהגדרת הפונקציה

print\_full\_name("lea ","hanuka")

**יודפס:** lea hanuka

**שימו לב:** התיעוד כולל גם הסבר אודות הפרמטרים שמקבלת כל פונקציה. ניתן להשתמש במונח פרמטר או ארגומנט לתאר את אותו הדבר: מידע המועבר לפונקציה. ניתן גם לדקדק ולומר ש: Args זהו קיצור של "ארגומנטים". ארגומנט הוא המשתנה כפי שהוא נקרא ע"י מי שקורא לפונקציה. פרמטר הוא המשתנה כפי שהוא נקרא בתוך הפונקציה.

**ההוראה return**

פונקציה יכולה להחזיר ערך או מספר ערכים ע"י ההוראה return.

**לדוגמא:**

def seven():

""" return 7 """

x=7

return x

a = seven()

print(a)

**יודפס:** 7

**דוגמא נוספת:**

def seven\_eleven():

""" return 7 & 11 """

x=7

y=11

return x,y;

var1,var2 = seven\_eleven()

print (var1,var2)

**יודפס: 11** 7

* המילה השמורה return מציינת את הוראת החזרה מהפונקציה (מחזירה ערך ויוצאת מהפונקציה), במקרה שהפונקציה מחזירה ערך. פקודת return לא חייבת להיות הפקודה האחרונה במתודה. במקרה כזה - המתודה מסתיימת מבלי לבצע את הפקודות הבאות לביצוע.

**דוגמה:**

def test(g):

if g>90:

return "Excellent"

elif g==90:

return "very good"

elif g>=80:

return "good"

print(test(95))

**יודפס:**

Excellent

**הערך None**

**דוגמה:**

def stam():

return None

s = stam()

print (s)

**יודפס:** None

הערך Noneאו "כלום", הוא ערך ריק. זוהי מילה שמורה בפייתון. משתנה יכול להיות שווה לערך ריק, ואז הוא שווה .None

**יש 3מצבים בהם פונקציה מחזירה ערך :None**  
- לפונקציה אין return כלל.  
- לפונקציה יש return אבל בלי שום ערך או משתנה. הכוונה לשורה שבה כתוב רק .return   
- לפונקציה יש return None.

**scope (תחום הכרה) של משתנים**

**מה יקרה כתוצאה מהרצת הקוד הבא?**

def speak():

word= 'hi'

print(word)

speak()

print(word) אף החוצה משום מה...

**יודפס** hi ואז התכנית "תעוף", כיון שיש שגיאת הרצה. המשתנה word לא מוכר מחוץ לתחום הפונקציה speak. ברגע שיצאנו מהפונקציה speak המשתנה word נמחק ואינו קיים יותר.

**מה יקרה כתוצאה מהרצת הקוד הבא?**

name ="kuki"

def speak():

word = 'hi'

print (word,name);

speak();

**התכנית תדפיס** hi kuki

**הסבר:** name הוא משתנה גלובלי – משתנה שמוגדר מחוץ לכל הפונקציות ולכן הוא מוכר בכולן. כלומר ה - scopeשל nameהוא כל הסקריפט שלנו.

**מה יקרה כתוצאה מהרצת הקוד הבא?**

word = 'bye'

def speak():

word = 'hi'

print(word)

speak();

**התכנית תדפיס** hi.

**הסבר:** בפונקציה speakאנחנו"דורסים" את המשתנה הגלובלי word עם משתנה לוקלי בעל אותו שם. כעת כאשר נזמן את ,speak הוא כבר לא יכיר את המשתנה הגלובלי, אלא רק את מה שהוגדר לוקלית.

**מה יקרה כתוצאה מהרצת הקוד הבא?**

word = 'bye'

def speak():

word = 'hi'

print(word)

speak();

print(word)

**יודפס:**

hi

bye

**הסבר:** את ההדפסה של hi כבר הבנו. כאשר speak מסיימת את הריצה שלה, המשתנה הלוקלי word נמחק, וכעת קורה משהו מעניין – מסתבר שהמשתנה הגלובלי word לא נמחק, אלא פשוט נשמר בצד. לפייתון יש מדרג של עדיפויות: כאשר פונים למשתנה בתוך פונקציה, קודם כל פייתון מחפש אם קיים משתנה לוקלי כזה, ולאחר מכן אם קיים משתנה גלובלי. ההדפסה השנייה מדפיסה את המשתנה הגלובלי .word

**מה ידפיס הקוד הבא?**

def speak(word):

word += 'luck'

print (word)

return word

word = 'good '

word = speak(word)

print(word)

**יודפס:**

good luck

good luck

**הסבר:**

זו דרך לשנות משתנה ע"י פונקציה. לקבל אותו כפרמטר ולהחזיר אותו באמצעות return לקוד שקרא לו.

**כאשר אנו לא יודעים מראש כמה פרמטרים הפונקציה תקבל**

**דוגמה: מה ידפיס הקוד הבא?**

def my\_function(\*kids):

print("Children names: " , kids)

my\_function("Ami", "Tami", "Rami")

my\_function("Bari", "Tali", "Shali", "Danny")

**יודפס:**

Children names: ('Ami', 'Tami', 'Rami')

Children names: ('Bari', 'Tali', 'Shali', 'Danny')

**תרגיל:**

במסגרת לימוד התכונות של הפונקציה חזקה, מעניין לבחון את היחס בין XY ל- YX.

עבור שני מספרים שלמים חיוביים שונים X ו- Y. עבור אלו ערכים מתקיים השוויון YX=XY ? ומהו היחס בין XY ל- YX כאשר X קטן ו- Y גדול ( למשל ערכו של איזה מספר גדול יותר 210 או 102)?

כתבו תכנית אשר קולטת שני מספרים שלמים וחיוביים X ו – Y ושולחת אותם לפונקציה המחשבת פעם את XY ופעם את YX (#example14)

def power(base,exponent):

p=1

for i in range(1,exponent+1):

p\*=base

return p

x = int(input("please enter an integer number "))

y = int(input("please enter an integer number "))

print("power(x,y)=", power(x,y))

print("power(y,x)=", power(y,x))

**דוגמאות לזימון פונקציות מסוגים שונים**

**פונקציות שמחזירות ערך:**

**כותרת פונקציה המחזירה ערך מספרי:**

def numOfDigits (num):

**אפשרות זימון – בפעולת פלט**:

print (" number of digits in " ,number ," : ", numOfDigits(number))

**כותרת פונקציה המחזירה ערך בוליאני:** #example15

def isPositive(num):

**אפשרות זימון – במשפט תנאי**:

#if isPositive(num1)==True and isPositive(num2)==True:

if isPositive(num1) and isPositive(num2):

print("the numbers:",num1,num2,"are Positive")

**כותרת פונקציה המחזירה ערך מספרי:**

def sumOfMoney (payment1 , payment2):

**אפשרות זימון – תוך הצבה במשתנה:**

sum = sumOfMoney (pay1 , pay2)

print(" Total amount of payment: ", sum)

**פונקציה שאינה מחזירה ערך:**

**כותרת הפונקציה:**

def displa*y (num*, ch):

**זימון הפונקציה:**

display ( n,sign);