**עבודת הגשה מס' 2**

תאריך הגשה הוא 12/12/2019

* ניתן להכין את המטלה בזוגות. רק חבר אחד בצמד יגיש בפועל את העבודה (במידה ומוגש כעבודה זוגית, יש לרשום בהערה את שמות המגישים ואת מספרי הזהות שלהם). יש להגיש את קבצי הפתרון תחת שם המכיל את מספרי ת"ז של המגישים.
* את החלק התיאורטי יש להגיש בפורמט PDF ואת החלק המעשי יש להגיש בקובץ נפרד בפורמט PY.
* חובה להשתמש בשמות הפונקציות המוגדרות.
* שימו לב, הפלט של דוגמאות ההרצה הוא בהתאם לסביבת הפיתוח Python IDLE ( בהרצה מתוך הscript ).
* חובה לכל פונקציה להוסיף doc strings.
* הגשה דרך מודל בלבד!
* כל שאלה בנוגע לתרגיל יש להפנות אך ורק לאחראי על התרגיל - ילנה באימייל: [**elena.chk@gmail.com**](mailto:elena.chk@gmail.com) . פניות בכל בדרך אחרת – לא יענו! בפנייה, יש לציין את : שם הקורס ופרטים מזהים.
* אישורי ההארכה יינתנו ע"י מרצה בלבד !

\* שימו לב: קיים הבדל עקרוני בין הדפסה לבין החזרה של ערך מפונקציה! ברירת המחדל בהיעדר הוראת הדפסה מפורשת היא החזרה בלבד.

**חלק א: דיאגרמות סביבה**

1. צייר דיאגרמת סביבה עבור הפעלת פונקציה הבאה:

def func1(x):

def funcA(y):

if abs(y\*\*2-x)<0.001:

return True

else:

return False

def funcB(y):

while funcA(y):

y=funcC(y)

return y

def funcC(y):

return (y+x/y)/2.0

return funcB(1.0)

func1(4)

1. צייר דיאגרמת סביבה עבור הפעלת פונקציות הבאות:

def f(x):

return 2\*x + 3

def g(f, x, y):

x = f(x) + f(y)

return x

x = 10

g(f, x, x\*2)

1. צייר דיאגרמת סביבה עבור הפעלות פונקציה הבאה. ציין מה יופיע בפלט. (שימו לב שיש מספר הרצות שונות):

a = 0

def f1(x):

def f2(y):

nonlocal x

x = x \* 2

return x > y

def f3(y):

global a

a = a + 1

return a + y

def f4(y):

nonlocal x

x = x + 1

return y + x

y = 'a'

if f2(x):

y = f3(x)

print('a={0}, x={1}, y={2}'.format(a, x, y))

return f4;

f1(5)(10)

f1(10)

f1 = f1(5)

print(f1(10))

print(f1(10))

**חלק ב: פונקציות מסדר גבוה ופונקציות ללא שם (lambda)**

1. כתבו פונקציה **Make\_iterator** אשר בהינתן פונקציה fn תחזיר פונקציה חדשה שתעבוד ב-iterator. Iterator מאפשר למתכנת לעבור על כל האיברים באוסף ללא קשר למימוש.

דוגמאות הרצה :

>>> fn= lambda n: 2\*n # Returns the n-th element in a sequence of even non-

# negative numbers (sequence starts from 0).

>>> iterator = **Make\_iterator**(fn) # Returns function that iterates over even non-

# negative numbers

>>> for i in range(4): # Prints four even numbers starting from the BEGINNING

print(iterator())

0

2

4

6

>>> for i in range(2, 4): # Prints NEXT two even numbers

print(iterator())

8

10

>>> for i in range(2, 4): # Prints NEXT two even numbers

print(iterator())

12

14

>>> it = **Make\_iterator**(fn) # creates a NEW iterator over even non-negative

# numbers

>>> for i in range(2): # Prints two even numbers starting from the BEGINNING

print(it())

0

2

>>> for i in range(2, 4): # Prints NEXT two even numbers in iterator

print(iterator())

16

18

1. לעיתים נרצה להפעיל פונקציה על סדרה של איברים. לשם כך נשתמש בפונקציה מסדר גבוה המתפעלת פונקציות אחרות ורצף של נתונים (sequence). כתוב את הפונקציות הבאות:
2. פונקציה listFilter(list, f) אשר מקבלת רצף (מסוג list) של מספרים ופונקציה כלשהי (מטיפוס בוליאני) ומחזירה רשימה מתוקנת שכוללת רק מספרים שהפונקציה מחזירה עבורם אמת.
3. פונקציה listFilterMulti(list, fList) אשר מקבלת רצף של מספרים ורצף של פונקציות ומפעילה סינון על סמך אותן פונקציות על המספרים ברצף. הפונקציה תחזיר רשימה מתוקנת. הסינון יתבצע על סמך עמידה בכל התנאים יחד (וגם).
4. פונקצית עזר isPrime(x) אשר תחזיר True אם מספר ראשונה ו-False אחרת.
5. פונקצית עזר isFib(x) אשר תחזיר True אם המספר הוא איבר בסדרת פיבונצ'י ו-False אחרת.

**דוגמה**:

הפעלה הבאה: listFilter([2,4,5,6,7,13], isPrime) תחזיר רשימה [13 ,7 ,5 ,2].

הפעלה הבאה: listFilterMulti([2,4,5,6,7,13], [isPrime, isFib]) תחזיר רשימה [13 ,5 ,2].

1. נקודה x נקראת נקודת שבת של פונקציה f אם f(x)=x. לפונקציות מסוימות ניתן לחשב נקודת שבת בשיטת "שיפור איטרטיבי", עם ניחוש התחלתי x0 והפעלה חוזרת של f:

f(x0), f(f(x0)), f(f(f(x0))), …

עד ההתכנסות (כלומר, עד שמספר לא משתנה בהרבה).

כתוב פונקציה בשם **Fixed\_point**אשר מקבלת פונקציה וניחוש התחלתי כארגומנטים ומחשבת את נקודת השבת בשיטת "שיפור איטרטיבי". פונקציה צריכה להחזיר נקודת שבת אם ניתן לחשב ו-None אם לא ניתן.

שימו לב: לא לכל פונקציה ניתן לחשב נקודת שבת בשיטה הזאות.

לדוגמא:

* אם נחשב נקודת שבת עבור f(x)= החל מ-4, נקבל סדרת המספרים שמתכנסת ל-1:

2 , 1.4142… , 1.1892… , 1.0905… , …

>>> **Fixed\_point**(lambda x: sqrt(x), 4)

תחזיר מספר קרוב ל-1 (תלוי מהסף שבחרתם לבדיקה).

* אך אם נחשב נקודת שבת עבור f(x)= החל מ-4, נקבל סדרה שלא מתכנסת:

16, 256, 65536, …

>>> **Fixed\_point**(lambda x: x\*\*2, 4)

צריכה להחזיר None במקרה כזה.

<https://en.wikipedia.org/wiki/Fixed_point_(mathematics)> הסבר נוסף

**בהצלחה !**