**עבודת הגשה מס' 3**

**תאריך הגשה – 16/12/2021**

**בעבודה זו חל איסור להשתמש בפתרונות המבוססים על נושאים שטרם נלמדו.**

* 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 ZIP עם שם קובץ מס' ת.ז. (לדוגמא אם מס' ת.ז. 123456789, קובץ להגשה 123456789[[1]](#footnote-1).zip) הכולל קובץ PY עם שם "solution.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: Data abstraction, Immutable data**

1. יש להגדיר טיפס שלא ניתן לשנות (**immutable type**) של מעגל )(**make\_circle(x, y, radius**). המעגל מיוצג ע"י המיקום של מרכז המעגל (**x, y**) ורדיוס.

המימוש חייב ליישם את עיקרון של הפשטת נתונים (**data abstraction**). יש לממש פעולות הבאות בשכבות הפשטה שונות:

* 1. **x** - מקבלת מעגל ומחזירה קואורדינטה x של מרכז המעגל.
  2. **y** – מקבלת מעגל ומחזירה קואורדינטה y של מרכז המעגל.
  3. **radius** – מקבלת מעגל ומחזירה רדיוס של המעגל.
  4. **print\_circle** – מקבלת מעגל ומדפיסה אותו.
  5. **perimeter** – מקבלת מעגל ומחזירה את הקף של המעגל.
  6. **area** – מקבלת מעגל ומחזירה שטח של המעגל.
  7. **distance** – מקבלת שני מעגלים ומחזירה את המרחק בין המרכזים של שני המעגלים
  8. **replace** – מקבלת מעגל וקואורדינטות חדשות של מרכז המעגל. מחזירה מעגל חדש עם מרכז המעגל מעודכן.
  9. **set\_radius** – מקבלת מעגל ורדיוס חדש. מעדכנת את הרדיוס מחזירה מעגל **חדש** אחרי השינויים.
  10. **add** – מקבלת שני מעגלים ומחזירה מעגל חדש המתקבל מסכום של הרדיוסים של שני המעגלים וממקמת אותו באמצע בין שני המעגלים שקיבלה בקלט.

**הערה:** אין להשתמש בטיפוסים מובנים של Python (חוץ ממספרים, tuple ופונקציות).

**דוגמת הרצה מחייבת :**

>>> c1 = make\_circle(4,5,10)

>>> c1

<function make\_circle.<locals>.dispatch at 0x03A758E8>

>>> x(c1)

4

>>> y(c1)

5

>>> radius(c1)

10

>>> print\_circle(c1)

Circle: point = (4,5); radius = 10

>>> perimeter(c1)

62.83185307179586

>>> area(c1)

314.1592653589793

>>> distance(c1, make\_circle(7,4,5))

3.1622776601683795

>>> print\_circle(replace(c1, 6, -7))

Circle: point = (6,-7); radius = 10

>>> print\_circle(set\_radius(c1, 9))

Circle: point = (4,5); radius = 9

>>> print\_circle(replace(set\_radius(c1, 5), 1.5, 3.2))

Circle: point = (1.5,3.2); radius = 5

>>> c2 = make\_circle(4, 6, 7)

>>> print\_circle(add(c1,c2))

Circle: point = (4.0,5.5); radius = 17

>>> print\_circle(add(c2,add(c1,set\_radius(make\_circle(1,2,3),1))))

Circle: point = (3.25,4.75); radius = 18

1. יש להגדיר טיפס שלא ניתן לשנות (**immutable type**) של תאריך )(**make\_date(d, m, y**).

המימוש חייב ליישם את עיקרון של הפשטת נתונים (**data abstraction**). יש לממש פעולות הבאות בשכבות הפשטה שונות:

* 1. **day** – מקבלת תאריך ומחזירה את היום.
  2. **month** – מקבלת תאריך ומחזירה את החודש.
  3. **year** – מקבלת תאריך ומחזירה את השנה.
  4. **print\_date** – מקבלת תאריך ומחרוזת (‘mdy’, ‘ymd’ או ‘dmy’) ומדפיסה את התאריך בהתאם לפורמט המתואר ע"י המחרוזת. למשל אם מחרוזת היא ,mdy התאריך יודפס בפורמט הבא: month/day/year.
  5. **next\_date** – מקבלת תאריך ומחזירה תאריך הבא. (אפשר להניח שבחודש פברואר יש תמיד 28 יום).
  6. **set\_day** – מקבלת תאריך ויום ומחזירה תאריך **חדש** עם יום מעודכן. (אפשר להניח שבחודש פברואר יש תמיד 28 יום).
  7. **set\_month** – מקבלת תאריך וחודש ומחזירה תאריך **חדש** עם חודש מעודכן.
  8. **set\_year** – מקבלת תאריך ושנה מחזירה תאריך **חדש** עם שנה מעודכנת. השנה המותרת היא בין 1900 לבין 2050.
  9. **diff** – מקבלת שני תאריכים ומחזירה פער בין תאריכים באותה שנה ואותו חודש בלבד, אחרת להחזיר -1

**הערות:** אין להשתמש בטיפוסים מובנים של Python (חוץ ממספרים, רשימות, tuple ופונקציות).

**דוגמת הרצה מחייבת :**

>>> d = make\_date(23,6,2021)

>>> d

<function make\_date.<locals>.dispatch at 0x034858A0>

>>> day(d)

23

>>> month(d)

6

>>> year(d)

2021

>>> print\_date(d,'mdy')

6/23/2021

>>> print\_date(d,'dmy')

23/6/2021

>>> print\_date(d,'ymd')

2021/6/23

>>> print\_date(d,'ydm')

Error date format!

>>> print\_date(set\_year(d, 1890), 'dmy')

Error year

>>> print\_date(set\_year(d, 2018), 'dmy')

23/6/2018

>>> print\_date(set\_month(d, 14), 'dmy')

Error month

>>> print\_date(set\_month(make\_date(30, 6, 2020), 2), 'dmy')

Day-Month inconsistency

>>> print\_date(set\_month(d, 11), 'dmy')

23/11/2021

>>> print\_date(set\_day(d,0), 'dmy')

Error day

>>> print\_date(set\_day(d,31), 'dmy')

Error day

>>> print\_date(set\_day(d,30), 'dmy')

30/6/2021

>>> print\_date(set\_year(set\_month(set\_day(d,15),7),2015),'dmy')

15/7/2015

>>> print\_date(next\_date(make\_date(25, 7, 2021)),'dmy')

26/7/2021

>>> print\_date(next\_date(make\_date(31, 7, 2021)),'dmy')

1/8/2021

>>> print\_date(next\_date(make\_date(31, 12, 2021)),'dmy')

1/1/2022

>>> print\_date(next\_date(next\_date(next\_date(make\_date(27,2,2010)))),'dmy')

2/3/2010

>>> diff(make\_date(26,4,1999),make\_date(15,4,1999))

11

**חלק ב: Conventional Interface, Pipeline**

1. בכל משימות הנתונות בסעיף זה יש להשתמש בפונקציות מובנות שנלמדו בכיתה: map, filter, reduce, וכד'. כל הפונקציות שתכתבו בתרגיל זה צריכות לתמוך בכל רצף ש-Python תומך בו, כלומר, כל רצף שהפונקציות לעיל תומכות בו או שלולאת for יודעת לעבור עליו.

**הערה: אסור להשתמש בלולאות בשאלה הנ"ל, כולל for בכל צורה שהיא.**

1. לכתוב פונקציה convert כך שבהינתן רשימה של מספרים שלמים, הפונקציה תבדוק עבור כל אחד מהמספרים ברשימה האם המספר הוא ציון תקין בין 0 ל-100. אם כן, הפונקציה ,תשייך אותו לאחד הקטגוריות הבאות:

* A אם המספר הוא בין 91 ל-100.
* B אם המספר הוא בין 81 ל-90.
* C אם המספר הוא בין 71 ל-80.
* D אם המספר הוא בין 56 ל-70.
* Fail אם ציון נמוך מ-56.

**דוגמת הרצה:**

>>> lst = (20, 45, 90, 3, 68, -30, 81, 98, 104, 63, 61)

>>> print(convert(lst))

(('A:', [98]), ('B:', [90, 81]), ('D:', [68, 63, 61]), ('Fail:', [20, 45, 3]))

הערה: שימו לב שקטגוריה C אינה מודפסת מכיוון שהיא ריקה.

1. לכתוב פונקציה factכך שבהינתן רשימה של רשימות של הציונים ופקטור, הפונקציה תכפיל את כל הציונים בכל הרשימות בפקטור. במידה וציון אחרי הפקטור יהיה גדול מ-100 אז הציון הסופי יהיה 100. עבור כל אחד מהרשימות של הציונים שהתקבלו אחרי הפקטור, הפונקציה תחזיר ציון מינימלי, ציון מקסימלי והממוצע.

**דוגמת הרצה:**

>>> lst = ((30, 80, 72, 40), (24,), (88, 50, 34, 90, 56))

>>> print(fact(lst,1.5))

((45.0, 100, 76.25), (36.0, 36.0, 36.0), (51.0, 100, 82.0))

הערה: אפשר להשתמש בפונקציות min, max, sum ו-len מובנות.

**חלק ג: Mutable data, message passing, dispatch function, dispatch dictionary**

1. יש לממש טיפוס נתונים חדש בשם **currency** שמייצג שער יציג ע״י סכום וסימון של אותו מטבע תוך שימוש ב dispatch function ו message passing-. יש לממש את הפעולות הבאות:
2. גישה לערך הסכום וסימון של המטבע בהתאם להודעה ‘get\_value’
3. עדכון של ערך הסכום וסימון המטבע המתאים ‘set\_value’
4. ייצוג טקסטואלי. יש לייצג את הסכום ביחד עם הסימון דרך ההודעה המתאימה - ‘str’
5. החלפת מטבע תבוצע ע״י שליחה של סימון חדש ופונקציית lambda שעל ידו תמירו את ערך המטבע הקודמת.

הערה: אין להשתמש בטיפוסים מובנים של Python !!!

דוגמת הרצה:

>>> c = make\_currency(10.50, ‘$’)

>>> c(‘get\_value’)(‘amount’)

*10.50*

>>> c(‘get\_value’)(‘symbol’)

*‘$’*

>>> c(‘set\_value’)(‘amount’, 50)

>>> c(‘get\_value’)(‘amount’)

*50*

>>> c(‘str’)

*'$50.00'*

>>> c(‘convert’)(lambda x: x\*3.87, ‘₪’)

>>> c(‘str’)

*‘₪193.50’*

1. בשאלה זו אתם מתבקשים לממש, ע"י השלמה, טיפוס נתונים חדש בשם **make\_medical\_Record** שמייצג תיק רפואי בחדר מיון של בית חולים תוך שימוש ב-**dict**, ב-**dispatch dictionary** ו-**message passing**. בתיק רפואי רשומים כל התחנות שעובר מטופל (מחרוזת) יחד עם זמנים (מחרוזת בפורמט '**hh**:**mm**') שמאוחסנים במלון. יש לממש את הפעולות הבאות: בדיקה אם מטופל עבר תחנה – **inData**, בדיקה אם מטופל לא עבר תחנה – **notInData**, הוספת תחנה שעבר מטופל לתיק רפואי – **addData**, הצגת זמנים (בסדר עולה) שמטופל עבר בתחנה – **view** והדפסה כל התחנות כולל זמנים שעבר מטופל הממוינות לפי סדר עלה של זמנים בהתאם לדוגמת הרצה – **printRecord**.

**דוגמת הרצה:**

>>> mr=**make\_medical\_Record**('David',1)

>>> mr

{'inData': <function make\_medical\_Record.<locals>.inData at 0x0425E808>, 'notInData': …, 'addData': …, 'view': …, 'printRecord': …}

>>> mr[**'addData'**]('11:30','registration')

>>> mr[**'inData'**]('registration')

True

>>> mr[**'inData'**]('registration')

True

>>> mr[**'inData'**]('procedure')

False

>>> mr[**'notInData'**]('procedure')

True

>>> mr[**'addData'**]('12:50','doctor checkup')

>>> mr[**'addData'**]('11:40','doctor checkup')

>>> mr[**'addData'**]('12:40','procedure')

>>> mr[**'addData'**]('14:30','doctor checkup')

>>> mr[**'addData'**]('13:30','radiography')

>>> mr[**'addData'**]('13:40','blood test')

>>> mr[**'view'**]('doctor checkup')

11:40,12:50,14:30,

>>> mr[**'view'**]('hospital discharge')

no events

>>> mr[**'addData'**]('15:00','hospital discharge')

>>> pr=mr[**'printRecord'**]()

David 1

>>> pr

{'hasMore': <function make\_medical\_Record.<locals>.printRecord.<locals>.hasMore at 0x0425E970>, 'next': …}

>>> pr[**'next'**]()

'11:30-registration'

>>> while pr[**'hasMore'**]():

pr[**'next'**]()

'11:40-doctor checkup'

'12:40-procedure'

'12:50-doctor checkup'

'13:30-radiography'

'13:40-blood test'

'14:30-doctor checkup'

'15:00-hospital discharge'

**חלק ד: שאלות טאורטיות**

1. סמנו אילו מהטענות נכונות והסבירו בקצרה לכל טענה:
   1. בפייתון ניתן להגדיר שני משתנים שמצביעים לאותו אובייקט.
   2. בפייתון לא ניתן להגדיר שני משתנים שמצביעים לאותה פונקציה.
   3. פונקציה ללא שם אחת יכולה להחזיר אובייקטים מטיפוסים שונים לפי תנאי.
   4. פונקציה ללא שם יכולה להכיל יותר משורת קוד אחת.
   5. לפונקציות מובנות כמו map אפשר להעביר כארגומנט רק פונקציות ללא שם.
   6. מילון (dictionary) הוא מבנה גמיש לחלוטין , שאין שום מגבלה על הטיפוסים של אובייקטים.
   7. לפי שיטת dispatch function עם message passing פונקציה dispatch יכולה לקבל רק פרמטר אחד בלבד.
   8. רשימה רקורסיבית (rlist) שמימשתם בכיתה (כפונקציה dispatch) היא רצף.

**בהצלחה !**

1. 2 מס' ת"ז יש לכתוב בשם הקובץ דרך קו (דוגמה: 123456789-987654321.zip) [↑](#footnote-ref-1)