המכללה האקדמית להנדסה סמי שמעון

עבודת הגשה מס' 4

תאריך הגשה הוא 23/01/2020

- ✓ 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. קובץ PY יכלול את השם "solution.py". את כל הקבצים יחד יש להגיש בקובץ ZIP עם שם קובץ מס' ת.ז (למשל, 123456789.zip).
- ✓ 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 - מאי חג'בי, באימייל: mayhagbi@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.

המכללה האקדמית להנדסה סמי שמעון

חלק 1: OOP (Python) ומימוש מערכת אובייקטים (Shmython)

1. ממש את המחלקות הבאות ב-Python :
   - א **Date** - המייצגת תאריך כולל פירוט של שנה, חודש ויום בחודש .
   - ב **Time** – המייצגת זמן כולל שעה ודקה .
   - ג **CalendarEntry** - מחלקה המייצגת רשומה בלו"ז הכוללת רשימת מטלות (עם זמן התחלה וזמן סיום) בתאריך מסוים. אין שתי מטלות עם בדיוק אותם זמנים!

<u>הערה 1</u>: יש להגדיר פונקציות גנריות repr ו str -על הטיפוס האלו.
<u>הערה 2</u>: יש להשתמש בצמד של מחרוזות המייצגות מופעים של מחלקת Time (פלט של מימוש __str__) בתור מפתח ייחודי של מטלה (task) (ראה דוגמת הרצה).
<u>הערה 3</u>: יש להדפיס מטלות ממוינות בסדר של זמן התחלה.

<u>דוגמת הרצה מחייבת (חלקית):</u>

```
>>> today = Date(2017, 1, 20)
>>> today
Date(2017, 1, 20)
>>> today.year
2017
>>> print(today)
‘20th of January, 2017’
>>> todo = CalendarEntry(2017, 1, 20)
>>> t = Time(10,0)
>>> str(t)
‘10:00’
>>> todo.addTask(“PPL lecture”, t, Time(13,0))
>>> todo.addTask(“PPL homework#4”, Time(14,0), Time(16,0))
>>> todo.tasks
{(’14:00’, ’16:00’): 'PPL homework#4', (‘10:00’, ’13:00’): 'PPL lecture'}
>>> print(todo)
Todo list for 20th of January, 2017:
1. 10:00-13:00 - PPL lecture
2. 14:00-16:00 – PPL homework#4
```

2. ממש את אותן מחלקות ב - Shmython:
   - א **Date** - ש לממש פונקציה make_date_class()
   - ב **Time** - יש לממש פונקציה make_time_class()
   - ג **CalendarEntry** - יש לממש פונקציה make_calentry_class()

<u>הערה 1</u>: יש לצרף קוד של מערכת אובייקטים (basic.py) שנבנה בכיתה להרצה.
<u>הערה 2</u>: אין לממש פונקציה , repr אך יש לממש את __str__ בטיפוס Time שתחזיר מחרוזת המייצגת זמן (לשימוש במילון של המטלות)
<u>הערה 3</u>: אין לתמוך בהדפסה של מטלות כמו בדוגמה הקודמת.

<u>דוגמת הרצה מחייבת (חלקית):</u>

```
>>> Date = make_date_class()
>>> today = Date[‘new’](2017, 1, 20);
>>> today[‘get’](‘year’)
2017
>>> CalendarEntry = make_calentry_class()
>>> todo = CalendarEntry[‘new’](2017, 1, 20)
>>> Time = make_time_class()
>>> t = Time[‘new’](10,0)
>>> t[‘get’](‘__str__’)()
‘10:00’
>>> todo[‘get’](‘addTask’)(“PPL lecture”, t, Time[‘new’](13,0))
>>> todo[‘get’](‘addTask’)(“PPL homework#4”, Time[‘new’](14,0), Time[‘new’](16,0))
>>> todo[‘get’](‘tasks’)
{(’14:00’, ’16:00’): 'PPL homework#4', (‘10:00’, ’13:00’): 'PPL lecture'}
```

המכללה האקדמית להנדסה סמי שמעון

חלק 2: פונקציות גנריות (generic functions)

3. יש לממש שלוש מחלקות המייצגות מטבע: Euro, Dollar, Shekel בבנק ישראל. כל מופע של מטבע יאותחל עם סכום במטבע המקורי, אך ניתן לקבל את ערכו בשקלים (amount) . פעולת חיבור (**add**) בין 2 מופעים שלהם תחזיר סכום שלהם בשקלים. יש להגדיר מילון עזר עם שערי המרה בין מטבעות חוץ לשקל. יש לממש את הפונקציות הנדרשות להדפסה ותצוגה של המפרש.

הערה: יש ליישם שיטת העמסת אופרטור (ממשק משותף)

דוגמת הרצה מחייבת:

```
<<< rates ={('dollar', 'nis'): 3.82,('euro','nis'): 4.07}
>>>s = Shekel(50)
>>>d = Dollar(50)
>>>e = Euro(50)
>>> d.amount()
191.18
>>>e.amount()
203.51
>>>d + s
241.18
>>>add(e, d)
394.69
>>>z=eval(repr(d))
>>>print(z)
50.0$
>>>print(s)
50.0nis
>>>print(e)
50.0€
```

4. יש לממש פונקציה גנרית apply המקבלת שם של פעולה, ושמות טיפוסים של ארגומנטים. הפונקציה תבצע את הפעולות חיבור וחיסור בין מטבעות שונים (אותם סוגים כמו בסעיף 3) ותחזיר כתוצאה מופע של מחלקה (מטבע) המייצג את הארגומנט הראשון (משמאל).

הערה1: יש ליישם את השיטה dispatch on type שנלמדה בהרצאה.
הערה2: ניתן להיעזר במילון rates המורחב.
דוגמת הרצה מחייבת:

```
>>>apply('add', Shekel(50), Dollar(20))
Shekel(131.4)
>>> rates[('euro','dollar')] = 1.06
>>>apply('add', Dollar(50), Euro(20))
Dollar(71.29)
>>>apply('sub', Dollar(50), Euro(20))
Dollar(28.71)
```

5. יש לממש פונקציה גנרית , coerce_apply המקבלת שם של פעולה ושמות טיפוסים של ארגומנטים (אותם טיפוסים כמו בסעיף 3), הפונקציה תבצע את הפעולה לאחר המרה של מטבע חוץ לשקלים ותחזיר כתוצאה מופע של המחלקה Shekel.

הערה1: יש ליישם את השיטה coercion שנלמדה בהרצאה.
הערה2: התוצאה המוחזרת תמיד תהיה מטיפוס Shekel .
דוגמת הרצה מחייבת:

```
>>> coercions[('dollar','nis')](Dollar(50))
Shekel(191.18)
>>> coerce_apply('add', Shekel(50), Dollar(20))
Shekel(131.4)
>>> coerce_apply('add', Dollar(50), Euro(20))
Shekel(272.73)
>>> coerce_apply('sub', Dollar(50), Euro(20))
Shekel(109.63)
```

המכללה האקדמית להנדסה סמי שמעון

חלק 3: חריגות (Exceptions)

6. שאלה זאת מתייחסת לשאלה **5** בעבודת בית מס' **3** (מימוש של **parking**).
יש לשדרג את המימוש שלכם ולהוסיף טיפול בחריגות (**ValueError**, **TypeError**, **IndexError**) של Python במקרים:

א) בהפעלת חניה (**parking**) יש לבדוק שהתשלום עבור שעת חניה הוא מספר חיובי ושיש מקום לחניה בכל אחד מסוגי החניונים.

ב) בכניסת רכב לחניה (**'start_parking'**) יש לבדוק אם ערך מספר הרכב לא תקין (לדוגמא מחרוזת) וגם את תקינות סוגי החניון (אין לקבל שם חניון שלא נמצא ברשימה אותם קיבלתם בעבודה 3).

ג) בהדפסת הפרטים של כל הרכבים הנמצאים בחניונים (**'print_list'**) לא יהיה צורך כעת בבדיקת סוף רצף **prn['end']()** אלא במקום, תצטרכו לבצע טיפול בחריגות.

<u>דוגמת הרצה (מחייבת):</u>

```
>>> park1=parking(-10,3,3,3)
the price value is bad
>>> park1=parking(10,0,3,3)
parking places error
>>> park1
>>> park1=parking(10,3,3,3)
>>> park1['start_parking']('aaa','VIP')
incorrect car number
>>> park1['start_parking'](223,'VIP1')
VIP1 is incorrect prking type
>>> park1['start_parking'](222,'Regular')
>>> park1['start_parking'](223,'Regular')
>>> park1['next_time']()
>>> park1['start_parking'](224,'Regular')
>>> park1['start_parking'](225,'VIP')
>>> prn=park1['print_list']()
>>> prn
{'next': <function parking.<locals>.print_list.<locals>.next at 0x03B6D618>}
>>> for _ in range(6):
prn['next']()
car: 222, parking type: Regular, parking time: 2
car: 223, parking type: Regular, parking time: 2
car: 224, parking type: Regular, parking time: 1
car: 225, parking type: VIP, parking time: 1
no car
no car
```

חלק 4: מבני נתונים רקורסיביים (recursive data structures)

7. אתם מתבקשים לממש מחלקה בשם Expr המייצגת ביטוי אריתמטי של ארגומנט אחד, כאשר ארגומנטים יכולים להיות ביטויים. המחלקה חייבת להכיל מימוש של **repr**. העלים יכולים להיות מכל טיפוס שונה מ-Expr.
8. יש לממש פונקציה ***build_expr_tree ,*** שבהינתן tuple שבנוי מ 3 אלמנטים (שם אופרטור במקום הראשון וארגומנטים במקום השני והשלישי) המייצג ביטוי אריתמטי, תבנה ותחזיר מופע של *Expr* עבור הביטוי.
הפונקציה חייבת להיות רקורסיבית!

<u>דוגמת הרצה:</u>

```
>>> exp = build_expr_tree(('add', ('mul', 2, 3), 10))
>>> exp
Expr('add', Expr('mul',3,2),10)
```

המכללה האקדמית להנדסה סמי שמעון

חלק 5: מפרש (Interpreter)

9. שאלה זו מתבסס על המפרש של מחשבון שמימשתם בכיתה.
אתם מתבקשים להרחיב/לעדכן את המפרש באופן הבא:
   א. אופרטור של כפל (**mul**) יקבל לפחות ארגומנט אחד.
   ב. אופרטור של חילוק (**div**) יחזיר inf (מספר אינסופי) במקרה של חלוקה ב-0 (ניתן לייצג בעזרת float("inf") ב-Python).
   ג. הרחיבו את המפרש עם אופרטור חדש בשם **round** שמקבל 2 מספרים (n ו-k) ומחזיר את המספר הראשון (n) מעוגל בעל k ספרות אחרי הנקודה. ניתן להשתמש בפונקציה המובנת של Python בשם round עם חתימה זהה.

<u>דוגמת הרצה מחייבת:</u>

```
>>> calc> mul()
TypeError: mul requires at least 1 argument
>>> calc> div(1, sub(2,2))
inf
>>> calc> mul(div(1, sub(2,2)),10)
inf
>>> calc> round(div(add(4,3), 3), 2)
2.33
>>> calc> round(div(add(4,3), 6), 2)
1.17
```

חלק 6: שאלות תיאורטיות

א במערכת אובייקטים שבנינו בכיתה (של Shmython) לא ניתן להוסיף פונקציות למאפיינים של מופע (instance attributes)

ב במפרש של מחשבון (calculator) שכתבנו בכיתה פונקציות **calc_eval** ו-**calc_apply** עובדות בצורת רקורסיה הדדית.

ג מטרת **memoization** היא לייעל חישוב רקורסיבי ע"י הקטנת ניצול זיכרון על חשבון זמן ריצה

ד ב- Python 3 משתמשים בהצהרה **nonlocal** על מנת לעדכן תוכן של מבנה נתונים (למשל, מילון) בסביבה כוללת (לא כולל מסגרת גלובאלית).

ה במפרש של מחשבון (calculator) שכתבנו בכיתה פונקציית **calc_eval** מממשת **רקורסית עץ** על מנת להעריך את הארגומנטים של הביטוי המוערך.

ו **פולימורפיזם בהורשה** ממומש בעזרת ממשק משותף בין הטיפוסים.

ז פונקציות גנריות (generic functions) המוגדרות על פרמטרים מטיפוסים זרים (**parametric polymorphism**) ממומשות ע"י הגדרת ממשק משותף בין הטיפוסים של ארגומנטים.

בהצלחה !