המכללה האקדמית להנדסה סמי שמעון

משימות למעבדה מס' 10

(Inheritance, Implementation classes and objects)

1) ממשו את המחלקה **ZoneTime** המרחיבה את המחלקה **Time**(שיצרתם במעבדה הקודמת) כך שתכיל גם את שם אזור הזמן (**Zone**).
בנוסף, עליכם להגדיר את הפונקציות הבאות:
א) פונקציית אתחול (יש להפעיל את הפונקציה המתאימה במחלקת האב).
ב) פונקציה המדפיסה את כל הפרטים כולל שם האזור.
ג) פונקציה המדפיסה את כל פרטי השעון, כולל שם האזור (דריסה של הפונקציה הווירטואלית!).
דוגמת הרצה (#הדפסה צפויה):

```
child=ZoneTime(10, 5, 34, 'Montreal')
zt=ZoneTime(10, 55, 34)  #10:55:34 \n Jerusalem
print('----------')
zt.printTime()
print('----------')
child.printTime()  #10:05:34 \n  Montreal
```

2) יש לממש מחלקה Point בשפת **Shmython** (דוגמה למימוש ב-python מצ"ב בקובץ lab_9_point_oop.py). כתוב דרייבר המתאים עבור המימוש הזה. יש להשתמש במה שנלמד בהרצאה לגבי מימוש אלטרנטיבי של מחלקות (dictionary) – הנמצא בקובץ Lab10_basic.py

3) יש להרחיב את המערכת כך שלכל מחלקה יהיה מאפיין של שם המחלקה וכן על כל מופע ידע לאיזו מחלקה הוא שייך.

לדוגמא:

```
>>> Point=make_point_class()
>>> P=Point['new'](0,0)
>>> P
{'set': <function make_class.<locals>.new.<locals>.set at 0x00F274F8>, 'get': <function
make_class.<locals>.new.<locals>.get at 0x00F27348>}
>>> P1=P['get']('class')['new'](1,2)
>>> P1
{'set': <function make_class.<locals>.new.<locals>.set at 0x00F27588>, 'get': <function
make_class.<locals>.new.<locals>.get at 0x00F27540>}
>>> P['get']('class_name')
'Point'
>>> Point['get']('class_name')
'Point'
>>> P1['get']('class_name')
'Point'
```

4) יש לשנות את המימוש הקיים , כך שלכל מחלקה יהיה מאפיין של מספר מופעים (שנוצרו אי פעם , ללא התחשבות באיסוף אשפה – garbage collection) של אותה מחלקה.

לדוגמא:

```
>>>Point=make_point_class()
>>>Point['get']('count')
0
>>>p=Point['new'](1,2)
>>>Point['get']('count')
1
>>>p=Point['new'](10,20)
>>>p=Point['new'](4,5)
>>>Point['get']('count')
3
```

5) יש להוסיף מחלקת *object*, כמו שיש ב-*Python*, כך שכל מחלקה שלא יורשת מאף מחלקה אחרת, תירש מה-*object* לפי ברירת מחדל (ולא מ-*None* כמו שזה במערכת המקורית). מחלקת *object* חייבת להכיל בנאי (*__init__*) ולכן, כתוצאת השינוי, תמיד יהיה לנו *__init__* בכל מחלקה. אחרי עדכון הנ"ל אפשר לדלג על בדיקת קיימות של בנאי ב-*make_class* :

```
init=get('__init__')
~~if init:~~
init(*args)
```

יש להשתמש במה שנלמד בהרצאה לגבי מימוש אלטרנטיבי של מחלקות (dictionary) – הנמצא בקובץ Lab10_basic.py

בהצלחה!

בהצלחה !