המכללה האקדמית להנדסה סמי שמעון

מטלה 4

מועד הגשה : 02/01/2019

- ✓ 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 הפיתרון 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 לאחראית על התרגיל – מרינה ליטבק באימייל: marinal@ac.sce.ac.il . פניות בכל בדרך אחרת – לא יענו!

  בפנייה, יש לציין את : שם הקורס, שם הקמפוס 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

שאלה 1

א. תגדיר מחלקות עבור:

- סטודנט לתואר ראשון (UndergraduateStudent),
- סטודנט לתואר שני (GraduateStudent),
- וסטודנט לתואר שלישי (PhDStudent).

לכל סטודנט מוגדר:

- שם (name),
- מחלקה שבה הוא לומד (dept),
- ושנת הלימודים בתואר הנלמד (year).

לכל סטודנט יש פונקציה **`introduce`** המדפיסה טקסט של הצגת הסטודנט כולל פרטים הבאים שלו:

- תואר הנלמד: Master, First ו-Ph.D. עבור תואר ראשון, שני ושלישי בהתאם,
- מחלקה,
- שם: סטודנט לתואר שני יוסיף לשמו מילה "Bachelor " (לפני השם) וסטודנט לתואר שלישי יוסיף ביטוי "Almost Doctor " לפני השם,
- ומס' שנים הנשאר להשלמת התואר: יש לחשב ע"י החסרה של שנת הלימודים ממשך של התואר הנלמד כאשר משך של תואר ראשון, שני ושלישי מוגדר כ-5, 4 ו-3 שנים בהתאם.

לכל טיפוס יש פונקציה **`str`** המדפיסה שם, תואר ומחלקה (dept) של הסטודנט ופונקציה **`repr`** המחזירה ייצוג טקסטואלי של המופע שיוצג ע"י המפרש: שם של מחלקה (class) וערכי מאפיינים (כולל פונקציות).

ניתן להגדיר מחלקות או\ו ממשקים נוספים במידת הצורך. ניתן לקבוע עץ הורשה לפי שיקול דעתך. עליך לדאוג שקוד של פונקציות ימומש גבוה ככל הניתן בהיררכית המחלקות ולא ישתכפל.

את הפתרון אפשר לתת או ע"י הורשה או ע"י הגדרת ממשק או פונקציות גנריות .

דוגמת הרצה:

```
>>> us = UndergraduateStudent("Moshe", "SE", 3)
>>> us.introduce()
I am a student for the first degree in SE department, my name is Moshe
and I will finish my studies in 2 years
>>> ms = GraduateStudent("Asaf", "CS", 2)
>>> ms.introduce()
I am a student for the Master degree in CS department, my name is
Bachelor Asaf and I will finish my studies in 2 years
>>> print(eval(repr(ms)))
MSc student Asaf from CS department
>>> ps = PhDStudent("Eli", "ISE", 1)
>>> ps.introduce()
I am a student for the Ph.D. degree in ISE department, my name is Almost
Doctor Eli and I will finish my studies in 2 years
```

ב. עליך לבדוק ולטפל בכל השגיאות האפשריות ע"י exceptions . יש לטפל בשגיות קלט הבאות:

- הכנסת מספר שלילי – יש להפעיל NegativeNumberError בעזרת הגדרת exception class
  - אם תוצאת חישוב של מספר השנים להשלמת התואר הוא מספר שלילי יש לזרוק NegativeNumberError
- הכנסת ערך לא תואם "טיפוס" הפרמטר (למשל, תו במקום מספר שלם) - יש להפעיל ValueError
- ניתן להשתמש ב ASSERT על מנת להפסיק ריצת התוכנית במקרה של קוד שגוי

ג. יש לממש את אותן מחלקות ב-SHMYTHON

<u>הערה 1</u>: אין צורך במימוש **repr** אך יש לממש את __str__.

<u>הערה 2</u>: שימו לב, שפונקציה **str** תופעל אך ורק ע"י הפעלה מפורשת של מתודת __str__.

<u>דוגמת הרצה:</u>

```
>>> us = UndergraduateStudent['new']("Moshe", "SE", 3)
>>> us['get']('introduce')()
I am a student for the first degree in SE department, my name is Moshe
and I will finish my studies in 2 years
>>> ms = GraduateStudent['new']("Asaf", "CS", 2)
>>> ms['get']('introduce')()
I am a student for the Master degree in CS department, my name is
Bachelor Asaf and I will finish my studies in 2 years
>>> print(ms['get']('__str__')())
MSc student Asaf from CS department
>>> ps = PhDStudent['new']("Eli", "ISE", 1)
>>> ps['get']('introduce')()
I am a student for the Ph.D. degree in ISE department, my name is Almost
Doctor Eli and I will finish my studies in 2 years
```

שאלה 2

א. יש לממש שלוש מחלקות המייצגות זמן: Week, Day, Hour . כל מופע של מחלקה תאותחל עם בכמות-ארגומנט של שבועות, ימים או שעות בהתאם, אך ניתן לקבל את ערכו במספר דקות (ע"י הפעלת מתודה **amount**)
פעולת חיבור (**add**) בין 2 מופעים שלהם תחזיר סכום שלהם בדקות.
יש לממש פונקציות הנדרשות להדפסה ותצוגה של מפרש.
<u>הערה</u>: יש ליישם שיטת העמסת אופרטור (ממשק משותף)

דוגמת הרצה מחייבת:

```
>>>s = Day(5)
>>>d = Week(2)
>>>e = Hour(50)
>>> d.amount()
20160
>>>e.amount()
3000
>>>d + s
27360
>>>add(e, d)
23160
>>>z=eval(repr(d))
>>>print(z)
2 weeks
>>>print(s)
5 days
```

ב. יש לממש פונקציה גנרית apply בהינתן שם של פעולה, ושמות טיפוסים של ארגומנטים. פונקציה תבצע את הפעולות חיבור וחיסור בין טיפוסים שונים ותחזיר תוצאה כמופע של מחלקה המייצגת את הטיפוס "הקטן" יותר.
הערה: יש ליישם שיטה dispatch on type

דוגמת הרצה מחייבת:

```
>>>apply('add', Day(50), Week(3))
Day(71)
>>>apply('sub', Week(1), Hour(20))
Hour(148)
```

ג. יש לממש פונקציה גנרית coerce_apply שבהינתן שם של פעולה ושמות טיפוסים של ארגומנטים תבצע את הפעולה לאחר המרה של זמן לשעות.
הערה: יש ליישם שיטה coercion
הערה: התוצאה תמיד תהיה מטיפוס Hour
דוגמת הרצה מחייבת:

```
>>> coercions[('day','hour')](Day(5))
Hour(120)
>>> coerce_apply('add', Hour(50), Day(2))
Hour(98)
>>> coerce_apply('add', Day(5), Week(2))
Hour(456)
```

שאלה 3

שאלה זאת מתבסס על מערכת תכנות מונחה עצמים שמימשנו בכיתה

א. שנה את המימוש הקיים, כך שלכל מחלקה יהיה מאפיין של שם המחלקה (**name**).
רמז: הוסף ארגומנט נוסף לפונקציה make_class שהוא שם המחלקה.

ב. שנה את המימוש הקיים, כך שלכל מופע יהיה מאפיין של טיפוס (**type**) שהוא השם של המחלקה המייצגת.

ג. שנה את המימוש הקיים, כך שלכל מופע יהיה מאפיין של המחלקה המייצגת (**this**).

ד. הוסף מתודה חדשה בשם ***mro*** שניתן להפעיל על כל מופע. המתודה צריכה להחזיר רשימת הטיפוסים (שמות של מחלקות) לפי סדר שבו מתבצע חיפוש של קוד המתודה שמופעלת על המופע (חוקי פולימורפיזם).

שים לב: יש לכתוב רק את השינויים, ולציין היכן הם מצויים בקוד של Shmython

דוגמת הרצה מחייבת:

```
<<> Account = make_account_class()
<<< Account['get']('name')
'Account'
>>> Jim = Account['new']('Jim')
>>> Jim['get']('type')
'Account'
>>> Bob = Jim['get']('this')['new']('Bob')
>>> Bob['get']('type')
'Account'
>>> SaveAccount = make_save_account_class()
>>> Jack = SaveAccount['new']('Jack')
>>> Jack['get']('type')
'SaveAccount'
>>> Jack['get']('mro')()
['SaveAccount', 'Account']
```

הערה חשובה : מחלקה SaveAccount יורשת ממחלקהAccount

שאלה 4

כתב פונקציה *accumulate_tree* שבהנתן פונקציה *f* (פעולה אריתמטית של שני ארגומנטים) ועץ (*tree* מיוצג כ- tuple) מחזירה את התוצאה המצטברת של *f* על כל העלים בעץ.
הפונקציה צריכה להיות רקורסיבית.

דוגמת הרצה:

```
>>> accumulate_tree(((1,2),3,4), add)
10
```

עבודה נעימה !