עבודת בית מס' 5:

<u>הנחיות כלליות:</u>

- קראו את כל ההוראות לגבי הגשת תרגילי הבית באתר הקורס.
- קראו את כל העבודה לפני שתתחילו לפתור אותה.
- תאריך פרסום: 14.06.22
- תאריך הגשת התרגיל: 27.06.22 בשעה 23:59.
- קובץ בדיקות נוסף ושאלת בונוס יעלו בסוף השבוע.
- יתאפשר איחורי הגשות בסך כולל של 5 ימים עבור כל התרגילים בסמסטר, לתרגיל שלא יוגש במסגרת זמן זה, יינתן ציון 0.
- כתבו תיעוד (הערות) שמסביר את הקוד שלכם. *<u>אסור לכתוב הערות בעברית!</u>*
- שאלות בנוגע לעבודה יישאלו בפורום המתאים במודל או בשעות הקבלה.
- את העבודה יש לכתוב בעורך הקוד המקובל בקורס (PyCharm) הגשת העבודה תתבצע באתר המודל בתיקייה הייעודית.
- **קבצי ההגשה יהיה קובץ מסוג py. במידה ויש יותר מקובץ אחד, על הקבצים להיות מרוכזים בתוך תיקיית zip. כששם התיקייה הוא תעודת הזהות של הסטודנט.**
- השימוש בחבילות מוכנות של פייתון **<u>אסור בהחלט</u>** ויגרור ציון 0.
- על הפלטים להיות **<u>בדיוק</u>** כפי שמוגדרים בשאלות (ללא רווחים מיותרים).

בדיקות ידניות:

לתרגיל זה מצורפת תכנית ובה כלל קבצי המחלקות בנוסף למחלקת ה-main, במחלקת ה-main נכללו מספר בדיקות בסיסיות עבור הבנאים והשיטות שתממשו בעבודה – שימו לב כי הבדיקות יעבדו רק לאחר מימוש כלל המחלקות בעבודה.

אין לשנות את חתימת המחלקות, התכונות והפונקציות המצוינות בתרגיל.

הגשת התרגיל:

תיקיית ההגשה ובה כל הקבצים הרלוונטיים של המחלקות תוגש במודל ותיהיה קובץ מסוג zip. כאשר שם הקובץ הינו תעודת הזהות של הסטודנט\ית. לדוגמא: 123456789.zip

1. **<u>כתבו מחלקת Subject המייצגת מקצוע בבית הספר לתלמיד מסוים בה נמצאות התכונות הבאות:</u>**

- תכונת name – שם המקצוע מסוג מחרוזת.
- תכונת grade – ציון המקצוע מסוג שלם (הציון הוא עבור תלמיד מסוים, עבור תלמיד נוסף – גם אם יש לו את אותו ציון יש ליצור אובייקט חדש).
- תכונת points – כמות נקודות זכות של המקצוע מסוג ממשי.

המחלקה תכיל את הפעולות הבאות:

- פעולת __repr__ - מחזירה (לא מדפיסה) מחרוזת המייצגת את המקצוע בצורה הבאה:

  'Subject:Algebra, grade:89, points:4.5'

- בנאי המקבל את שלושת התכונות לפי הסדר (קודם שם, אח"כ ציון ולבסוף נק"ז), אין צורך לבדוק תקינות נתונים (ציון יהיה תמיד שלם בין 0 ל 100, נק"ז יהיה בין 0.5 ל 5 בקפיצות של 0.5).

לדוגמא עבור הפקודות:

```
Algebra=Subject('Algebra',89,4.5)
Hedva=Subject('Hedva',67,5)
print(Algebra)
print(Hedva)
```

יודפס הפלט:

```
Subject:Algebra, grade:89, points:4.5
Subject:Hedva, grade:67, points:5
```

2. **<u>כתבו מחלקת Student המייצגת תלמיד תיכון שתכיל את התכונות הבאות:</u>**

- תכונת name – שם התלמיד מסוג מחרוזת.
- תכונת head – מקצועות בהם נבחן התלמיד, מצביע לחוליה ברשימה מקושרת של אברים מסוג Node (שימו לב לא מבנה הרשימה כולה אלא רק חוליה).
- תכונת year - שכבה המיוצגת על ידי מספר שלם בין 10 ל 12 המייצג כיתה בין י' ל-יב'.
- תכונת class_number – מספר שלם בין 1 ל 20 המייצג את מספר הכיתה בשכבה.

המחלקה תכיל את הפעולות הבאות:

- פעולת __repr__ - מחזירה (לא מדפיסה) מחרוזת המייצגת את התלמיד בצורה הבאה:
  'Student:uriel, class:Yod1, grades:Algera(4.5)-89, Hedva(5)-67.
  (עבור כיתה יא' יש לציין YodAlef ועבור יב' יש לציין YodBet).
- בנאי המקבל את התכונות לפי הסדר (שם, רשימת מקצועות – המקצוע באינדקס 0 יהיה ראש הרשימה וכך הלאה לפי הסדר, מספר שכבה ומספר כיתה), אין צורך לבדוק תקינות נתונים, ניתן להניח רשימת מקצועות ריקה, במקרה זה head יהיה None.
- פעולת get_average **רקורסיבית** (ללא לולאות וללא פונקציות עזר המכילות לולאות) המחזירה ממוצע אקדמאי של התלמיד (סכום מכפלת כל ציון בנק"ז חלקי סכום הנק"ז), עבור תלמיד ללא מקצועות יש להחזיר ממוצע 0.
- כל פעולות ההשוואה הנדרשות בכדי להשוות תלמידים על פי ממוצע אקדמאי (הפעולות הנדרשות להשוואות: == , < , > , <= , =>).

לדוגמא עבור הפקודות הבאות (בהמשך לפקודות הקודמות):

```
Algebra=Subject('Algebra',89,4.5)
Hedva=Subject('Hedva',67,5)
uriel=Student('uriel',[Algebra,Hedva],10,1)
yohai=Student('Yohai',[Subject('Algebra',98,4.5),Subject('Hedva',90,5)],12,3)
print(uriel.get_average())
print(uriel)
print(yohai)
print(uriel>=yohai)
print(uriel>yohai)
print(uriel<=yohai)
print(uriel<yohai)
print(uriel==yohai)
print(uriel!=yohai)
```

יודפס הפלט הבא:

```
77.42105263157895
Student:uriel, class:Yod1, grades:Algebra(4.5)-89, Hedva(5)-67.
Student:Yohai, class:YodBet3, grades:Algebra(4.5)-98, Hedva(5)-90.
False
False
True
True
False
True
```

3. **<u>כתבו מחלקת ClassRoom המייצגת כיתת בית ספר ויורשת ממחלקת מחסנית, המחלקה תכיל מעבר לירושה את התכונות הבאות:</u>**

- תכונת year – שכבה מסוג מספר שלם
- תכונת class_number – מספר כיתה מסוג מספר שלם

המחלקה תכיל את הפעולות הבאות (**יש "לדרוס" מינימום פעולות**):

- פעולת __repr__ המחזירה את המחרוזת של הכיתה בפורמט הבא:

'Class yod1, 25 students.'

- בנאי המקבל את מספר השכבה, מספר הכיתה ורשימת תלמידים (ניתן להניח שכולם אכן מתאימים לכיתה), התלמידים יוכנסו למחסנית הכיתה (שימו לב שהכיתה היא מחסנית מורחבת) החל מאינדקס 0 ועד לאינדקס המקסימלי.

לדוגמא עבור הפקודות הבאות (בהמשך לפקודות הקודמות):

```
yod1=ClassRoom(10,1,[uriel,Student('Yael',[],10,1)])
yod_bet_3=ClassRoom(12,3,[yohai])
print(yod1)
print(yod_bet_3)
```

יתקבל הפלט:

```
Class Yod1, 2 students.
```

Class YodBet3, 1 students.

4. כתבו מחלקת תור s_queue המייצגת תור ומכילה את הפעולות הבאות:

- enqueue(val) – הכנסת האיבר val לסוף התור.
- dequeue() – הוצאת האיבר הנמצא בראש התור והחזרתו. במידה והתור ריק, יש לזרוק שגיאה מסוג IndexError עם ההודעה – "The queue is empty"
- is_empty() – בדיקה אם התור ריק. אם כן, יש להחזיר True, אחרת – False.
- front() – החזרת האיבר הנמצא בראש התור מבלי להוציאו מהתור. במידה והתור ריק, יש לזרוק שגיאה מסוג IndexError עם ההודעה – "The queue is empty"
- rear() - החזרת האיבר הנמצא בסוף התור מבלי להוציאו מהתור. במידה והתור ריק, יש לזרוק שגיאה מסוג IndexError עם ההודעה – "The queue is empty"
- len(my_queue) - החזרת אורך התור.

אופן מימוש התור נתון להחלטתכם, אך וודאו כי אתם תומכים בכל הפעולות כנדרש!

לדוגמא עבור הפקודות הבאות:

```
q=s_queue()
q.enqueue('ma')
q.enqueue('kore')
q.enqueue('ah')
q.enqueue('shelo')
q.enqueue('gibor?')
print(q.front())
print(q.rear())
print(q.dequeue())
print(len(q))
print(q.front())
print(q.rear())
```

יודפס הפלט:

```
ma
gibor?
ma
4
kore
gibor?
```

5. **<u>כתבו מחלקת Year המייצגת שכבת בית ספר ומכילה את התכונות הבאות:</u>**

- תכונת year_number – שכבה מסוג מספר שלם
- תכונת classes_queue – תור מהסוג שהגדרתם בסעיף הקודם המכיל את כיתות השכבה

המחלקה תכיל את הפעולות הבאות:

- פעולת count_classes המחזירה את כמות הכיתות בשכבה (מספר שלם).
- פעולת __repr__ המחזירה את המחרוזת של השכבה בפורמט הבא:

'School year yod, 2 classes.'

- בנאי המקבל את מספר השכבה, ומערך כיתות, יש להזין את הכיתות לתור החל מאינדקס 0 ועד לאינדקס הגבוה ביותר לפי הסדר.

לדוגמא עבור הפקודות:

```
yod=Year(10,[yod1])
yod_alef=Year(11,[])
print(yod)
print(yod_alef)
```

יודפס הפלט:

```
School year Yod, 1 classes.
School year YodAlef, 0 classes.
```

6. **<u>כתבו מחלקת School המייצגת בית ספר ומכילה את התכונות הבאות:</u>**

- תכונת name – שם בית הספר מסוג מחרוזת
- תכונת school_years מסוג מערך (list) בגודל 3 המכיל את שלושת השכבות.

המחלקה תכיל את הפעולות הבאות:

- בנאי המקבל את שם בית הספר ורשימה של 3 שכבות (ניתן לעבוד עם העתקה רדודה, יש ברשימה בוודאות 3 שכבות המתאימות לכיתות י' עד יב').
- פעולת get_excellent המחזירה רשימה של שליש התלמידים המצטיינים בבית הספר לפי ממוצע אקדמאי, על התלמידים להיות ממוינים לפי הממוצע, הממוצע הגבוה ביותר יהיה באינדקס 0.

אם לא ניתן לחשב שליש במדויק יש לעגל כלפי מעלה, ניתן להניח ממוצע שונה לכל תלמיד (אם יש שני תלמידים בבית הספר ניתן להניח שאחד בלבד יהיה מצטיין, אם יש ארבעה אז שניים יהיו מצטיינים וכו').

בית ספר ריק (ללא תלמידים) יחזיר רשימה ריקה.

לדוגמא עבור הפקודות:

```
Algebra=Subject('Algebra',89,4.5)
Hedva=Subject('Hedva',67,5)
uriel=Student('uriel',[Algebra,Hedva],10,1)
yohai=Student('Yohai',[Subject('Algebra',98,4.5),Subject('Hedva',90,5)],12,3)
yuvali=Student('Yuvali',[Subject('Algebra',99,4.5),Subject('Hedva',99,5)],11,3)
hnana=Student('Hnana',[Subject('Algebra',100,4.5)],11,4)
yod1=ClassRoom(10,1,[uriel,Student('Yael',[],10,1)])
yod_bet_3=ClassRoom(12,3,[yohai])
yod_alef_3=ClassRoom(11,3,[yuvali])
yod_alef_4=ClassRoom(11,4,[hnana])
yod=Year(10,[yod1])
yod_alef=Year(11,[yod_alef_3,yod_alef_4])
yod_bet=Year(12,[yod_bet_3])
ort_arad=School('Ort_arad',[yod,yod_alef,yod_bet])
print(ort_arad.get_excellent())
```

יודפס הפלט:

[Student:Hnana, class:YodAlef4, grades:Algebra(4.5)-100., Student:Yuvali, class:YodAlef3, grades:Algebra(4.5)-99, Hedva(5)-99.]

<u>הערות:</u>

- בכדי שהתכנית תעבוד היטב יש לממש את מבני החוליה, מחסנית – לנוחיותכם הם מצורפים גם בסוף תרגיל זה וגם בקבצים המצורפים לעבודה.
- יש להקפיד על ורשימת שיטות של תור בדיוק כפי שהוצגו בתרגול וצוינו כאן.
- יש לעבוד עם פקודות מדויקות בכדי שהבדיקה תרוץ היטב.
- ניתן להניח תקינות קלט נתונים בכל הפעולות.
- השתמשו בבדיקות שמפורסמות בתרגיל בכדי לבדוק את הקוד שלכם.
- בהצלחה !!!!!

<u>נספח – מחלקת Node:</u>

```
class Node:
    def __init__(self, val):
        self.value = val
        self.next = None

    def __repr__(self):
        return '[' + str(self.value) + ']'
```

נספח – מחלקת Stack:

```
class Stack:
    def __init__(self):
        self.stack_vals = []
        self.len = 0

    def push(self, val):
        self.stack_vals.append(val)
        self.len += 1

    def pop(self):
        if self.is_empty():
            return None
        res = self.stack_vals[-1]
        self.stack_vals = self.stack_vals[0:-1]
        self.len -= 1
        return res

    def __repr__(self):
        out = '|'
        for i in range(self.len):
            out += str(self.stack_vals[i]) + ' '
        out += '<-top'
        return out

    def peek(self):
        if self.is_empty():
            return None
        return self.stack_vals[-1]

    def __len__(self):
        return self.len

    def is_empty(self):
        return self.len == 0
```

בהצלחה !!!! אוריאל.