תרגיל בית 8

הנחיות כלליות:

- קראו **בעיון** את השאלות והקפידו שהתכניות שלכם פועלות בהתאם לנדרש.
- את התרגיל יש לפתור לבד!
- הקפידו על כללי ההגשה המפורסמים באתר. בפרט, יש להגיש את כל הפתרונות לשאלות יחד בקובץ ex8_012345678.py המצורף לתרגיל, לאחר החלפת הספרות 012345678 שבשם הקובץ במספר תעודת הזהות שלכם, כל 9 הספרות כולל ספרת ביקורת.
- אופן ביצוע התרגיל: שימו לב, בתרגיל זה עליכם להשלים את הקוד בקובץ המצורף.
- **אין לשנות את שמות המחלקות, הפונקציות, המתודות והמשתנים שכבר מופיעים בקובץ השלד של התרגיל.**
- **אין למחוק את ההערות שמופיעות בקובץ השלד.**
- היות ובדיקת התרגילים עשויה להיות אוטומטית, **יש להקפיד על פלטים מדויקים על פי הדוגמאות (עד לרמת הרווח).**
- בדיקה עצמית: כדי לוודא את נכונותן ואת עמידותן של התוכניות לקלטים שגויים, בכל שאלה הריצו את תוכניתכם עם מגוון קלטים שונים, אלה שהופיעו כדוגמאות בתרגיל וקלטים נוספים עליהם חשבתם (וודאו כי הפלט נכון).
- **ניתן להניח שהקלט תקין בהתאם להערות המפורטות בהוראות כל שאלה, אלא אם מצויין אחרת.**
- מועד אחרון להגשה: כמפורסם באתר.

בתרגיל זה תכתבו תוכנה לניהול חדרי בית מלון ואורחיהם. התוכנה כוללת מחלקה המייצגת חדר במלון ומחלקה המייצגת את המלון. בנוסף מחלקה המייצגת מיניבר.

הערות כלליות חשובות:

- מומלץ לקרוא תחילה את כל התרגיל, להבין את הוראות התרגיל ולתכנן את המחלקות השונות והיחסים ביניהן בהתאם.
- במימוש המחלקות השונות יש להימנע משכפול קוד ככל שניתן!
- השוואת מחרוזות בכל התרגיל תיעשה לפי lowercase בלבד. לדוגמה, יש להתייחס למחרוזות "ABc" ו-"abC" כשוות, כיוון שבפורמט lowercase שתיהן הופכות למחרוזת "abc".
- בכל השאלות, ניתן להוסיף שדות ומתודות נוספים לאלה הנדרשים בשאלה, אם הדבר מסייע לכם במימוש הפיתרון.

השלימו את המחלקות בקובץ התרגיל בהתאם לדרישות המופיעות בשאלות שלהלן.

<u>שאלה 1</u>

א. ממשו את המחלקה Minibar. לכל מיניבר יש את התכונות (Attributes) הבאות:

שם	תיאור	סוג	הערות
drinks	המשקאות הנמצאים במיניבר.	מילון המכיל את השמות של המשקאות כמפתחות (keys) ואת מחירי המשקאות כערכים (values).	יכול להיות מילון ריק.
snacks	החטיפים הנמצאים במיניבר.	מילון המכיל את השמות של החטיפים כמפתחות (keys) ואת מחירי החטיפים כערכים (values).	יכול להיות מילון ריק.
bill	החשבון של המיניבר.	מספר עשרוני (float).	יש לאתחל כ- 0.

התחילו במימוש בנאי המחלקה על פי החתימה הבאה:

```
def __init__(self, drinks, snacks):
```

ניתן להניח שכל הקלטים תקינים, כמתואר בטבלה. אין צורך לבצע בדיקות של טיפוסים או תקינות הקלט.

ב. ממשו את המתודות eat_a_snack(self, snack) ו- drink_a_drink(self,drink).

מתודה eat_a_snack(self, snack) מקבלת שם של חטיף כמחרוזת ומסירה אותו מהמילון snacks. בנוסף, היא מוסיפה את המחיר של החטיף לחשבון (bill) של המיניבר.

מתודה drink_a_drink(self,drink) מקבלת שם של משקה כמחרוזת ומסירה אותו מהמילון drinks. בנוסף, היא מוסיפה את המחיר של המשקה לחשבון (bill) של המיניבר.

במידה והערך (שם החטיף או המשקה) אינו נמצא במילון המתאים, יש להעלות (raise) שגיאה מסוג ValueError עם ההודעה הבאה:

'The snack is not in the minibar' עבור חטיף

ו- 'The drink is not in the minibar' עבור משקה

ג. ממשו את המתודה __repr__(self) שמחזירה מחרוזת המתארת אובייקט מסוג Minibar, על פי דוגמת הפלט:

```
drinks1 = {'coke': 12, 'rum': 25}
snacks1 = {'m&m': 10, 'cake': 30}

m = Minibar(drinks1, snacks1)
print(m)
```

```
The minibar contains the drinks: ['coke', 'rum']
And the snacks: ['m&m', 'cake']
The bill for the minibar is: 0
```

<u>הערות:</u>

- יש להדפיס את הערכים בשורות נפרדות (\n) יחד עם הכיתוב המודפס בהדגמה.
- יש לכלול רווח בין הנקודתיים לערכים של כל attribute.
- השמות של החטיפים והמשקאות יודפסו כרשימה.
- אין צורך לכלול \n לאחר התכונה האחרונה.

דוגמת הרצה נוספת:

```
drinks1 = {'coke': 12, 'rum': 25}
snacks1 = {'m&m': 10, 'cake': 30}

m = Minibar(drinks1, snacks1)
print(m)



m.eat_a_snack('m&m')
print(m)
m.drink_a_drink('rum')
print(m)
```

```
The minibar contains the drinks: ['coke', 'rum']
And the snacks: ['m&m', 'cake']
The bill for the minibar is: 0
The minibar contains the drinks: ['coke', 'rum']
And the snacks: ['cake']
The bill for the minibar is: 10
The minibar contains the drinks: ['coke']
And the snacks: ['cake']
The bill for the minibar is: 35
```

ושגיאה:

```
drinks1 = {'coke': 12, 'rum': 25}
snacks1 = {'m&m': 10, 'cake': 30}

m = Minibar(drinks1, snacks1)
m.drink_a_drink('beer')
```

```
ValueError: The drink is not in the minibar
```

שאלה 2

א. ממשו את המחלקה **Room**, אשר מייצגת חדר בבית מלון. לכל חדר יש את התכונות (Attributes) הבאות:

שם	תיאור	סוג	הערות
minibar	המיניבר הממוקם בחדר	minibar שמומש בשאלה 1	מיניבר המכיל משקאות וחטיפים
floor	מספר הקומה בה נמצא החדר	מספר שלם int גדול או שווה לאפס	
number	מספר החדר	מספר שלם int חיובי ממש	
guests	רשימת שמות אורחי החדר הנוכחיים	רשימה של מחרוזות. המחרוזות ברשימה מכילות אך ורק אותיות (גדולות או קטנות) ורווחים	תיתכן רשימת אורחים ריקה (חדר ריק)
clean_level	רמת הניקיון של החדר בין 1 (מלוכלך) ל-10 (מבריק)	מספר שלם int בין 1 ל-10	
rank	דרגת החדר המייצגת את יוקרתיות החדר. ישנן 3 דרגות: 1 (בסיסית -Basic), 2 (סטנדרטית – Standard), או 3 (יוקרתית - Luxury)	מספר שלם int בין 1 ל-3	
satisfaction	רמת שביעות הרצון של אורחי החדר בין 1.0 (נמוכה ביותר) ל-5.0 (גבוהה ביותר)	מספר ממשי (float) בין 1.0 ל-5.0	* רמת שביעות הרצון בברירת מחדל (default) היא 1.0. * תיתכן רמת שביעות רצון שאינה שלמה, למשל: 4.5. * במידה והמשתמש הזין רמה שביעות רצון מסוג int, יש להמירה לfloat בעת שמירת כתכונה באובייקט.

התחילו במימוש בנאי המחלקה על פי החתימה הבאה:

__init__(self, minibar, floor, number, guests, clean_level, rank, satisfaction=1.0)

ניתן להניח את תקינות סוגי וערכי הקלטים כפי שמתואר בטבלה למעלה, פרט למקרים המתוארים להלן, שבהם יש לוודא תקינות:

- יש להתחיל בבדיקה של הטיפוסים. במקרה ואחד הטיפוסים לא תקין, יש להעלות (raise) חריג (Exception) מסוג TypeError. יש להתייחס למקרים הבאים:
  - רמת הנקיון אינה מסוג int.

- דרגת החדר אינה מסוג int.
- רמת שביעות הרצון אינה מספר שלם מסוג int וגם אינה מספר ממשי מסוג float. שימו לב: המשתמש יכול לבנות חדר עם רמת שביעות רצון שלמה. בפרט, ייתכן שהיא תינתן כארגומנט מסוג int. למשל, בתור הארגומנט 4 אשר מייצג רמת שביעות רצון 4.0.
- אין צורך לבדוק את טיפוס המיניבר. ניתן להניח שהוא תקין.

- לאחר בדיקת הטיפוסים נעבור לבדיקת ערכים. במקרה ואחד מהערכים אינו תקין (למרות שהטיפוס תקין), יש להעלות חריג מסוג ValueError. יש להתייחס למקרים הבאים:
  - רמת הניקיון אינה בין 1 ל-10 (כולל קצוות הטווח).
  - דרגת החדר אינה בין 1 ל-3 (כולל קצוות הטווח).
  - רמת שביעות הרצון אינה בין 1 ל-5 (כולל קצוות הטווח).

הערות:

- ניתן לבחור כרצונכם את ההודעה שבכל TypeError ובכל ValueError.
- במקרה שיש מספר ארגומנטים שונים שאינם תקינים, אין חשיבות מהו הארגומנט הבעייתי שבעקבותיו יועלה החריג.
- יש להמיר את כל האותיות בשמות האורחים ברשימה שמתקבלת כקלט ביצירת אובייקט חדש לאותיות קטנות (lowercase). אופן מימוש זה יקל על מימוש הסעיפים שבהמשך השאלה.

ב. ממשו את המתודה __repr__(self) שמחזירה מחרוזת המתארת אובייקט מסוג Room, על פי דוגמת הפלט שבהמשך.

- המחרוזת תכלול שורה נפרדת עבור כל אחת מתכונות החדר בפורמט "שם התכונה:<רווח אחד>ערך התכונה". מלבד למיניבר אשר יודפס בהתאם להגדרה בשאלה 1.
- סדר הופעת התכונות יהיה זהה לסדר הופעתן בטבלה למעלה.
- עבור התכונה guests, ערך התכונה יהיה שמות רשימת האורחים ב-lowercase (בסדר כלשהו), כאשר הינם מופרדים ב<<פסיק><רווח אחד>>. אם רשימת האורחים ריקה, ערך התכונה יהיה המילה empty (ראו דוגמה שנייה בדוגמאות הרצה).
- מכוון שתכונת satisfaction הינה מסוג float, היא תודפס למסך עד רמת דיוק של ספרה אחת אחרי הנקודה (עשו שימוש בround, ראו דוגמא)
- אין צורך לכלול \n לאחר התכונה האחרונה

דוגמאות הרצה:

'm' מייצג את המיניבר משאלה 1 בכל דוגמאות ההרצה

קלט:

```
m = Minibar({'coke': 10, 'lemonade': 7}, {'bamba': 8, 'mars': 12})
print(Room(m, 12, 101, ["Ronen", "Shir"], 6, 2))
```

פלט:

```
The minibar contains the drinks: ['coke', 'lemonade']
And the snacks: ['bamba', 'mars']
The bill for the minibar is: 0
floor: 12
number: 101
guests: ronen, shir
clean_level: 6
rank: 2
satisfaction: 1.0
```

ג. הוסיפו את מימוש המתודות הבאות למחלקה Room:

חתימת המתודה	תיאור
is_occupied(self)	מחזירה True אם החדר תפוס, כלומר, יש בחדר אורחים, ואחרת – False.
clean(self)	מבצעת פעולת ניקיון של החדר, שאיכותה ומידת השפעתה על רמת הניקיון עולה עם דרגת החדר. פעולת ניקיון אחת מעלה את רמת הניקיון של החדר $clean\_level$ ל-$\min(10, clean\_level + rank)$, כאשר $rank$ הוא דרגת החדר.
better_than(self, other)	משווה בין רמתם של שני חדרים, ומחזירה True אם self הוא חדר "יותר טוב" מהחדר other, ואחרת – False. • החדר self נחשב ל"יותר טוב" מהחדר other אם $(self.rank, self.floor, self.clean\_level) > (other.rank, other.floor, other.clean\_level)$ כאשר הסדר $>$ **הוא כפי שהוא מוגדר על tuples.** • אם other אינו מסוג Room, יש להעלות חריג מסוג TypeError עם ההודעה: "Other must be an instance of Room".
check_in(self, guests)	מכניסה אורחים לחדר. • המתודה תכניס את רשימת האורחים guests לחדר self אם הוא ריק, ובנוסף, תאתחל את רמת שביעות הרצון ל-1.0. • אם החדר תפוס, לא ניתן לבצע זאת, ולכן המתודה תעלה חריג מסוג RoomError עם ההודעה: "Cannot check-in new guests to an occupied room". • ניתן להניח ש-guests היא רשימת מחרוזות <u>לא ריקה</u> עם שמות חוקיים (כלומר, שכוללים רק אותיות אנגליות ורווחים), ואותיות שיכולות להיות בפורמט case גדול או קטן. • <u>הערה:</u> על המתודה להכניס את שמות האורחים לשדה self.guests בפורמט lowercase בלבד.

check_out(self)	מבצעת צ'ק-אאוט, כלומר, מפנה את האורחים השוהים כעת בחדר. • אם רשימת האורחים של החדר (self.guests) <u>אינה</u> ריקה, אז הפינוי כולל את הפיכתה לרשימה ריקה. • אחרת, אם self.guests ריקה, יש להעלות חריג מסוג RoomError עם ההודעה: "Cannot check-out an empty room".
move_to(self, other)	מעבירה את אורחי החדר self לחדר other אם הוא ריק. • אם self ריק, אין אורחים להעביר, ולכן המתודה תעלה חריג מסוג RoomError עם ההודעה: "Cannot move guests from an empty room". • אם other תפוס, לא ניתן לבצע את העברת האורחים, ולכן המתודה תעלה חריג מסוג RoomError עם ההודעה: "Cannot move guests into an occupied room". • אם self ריק וגם other תפוס, אז יש להעלות את החריג אודות self. <u>פעולת העברת האורחים מהחדר self כוללת את הצעדים הבאים:</u> א. העברת שמות האורחים מ-self לרשימה המתאימה ב-other. ב. אם other הוא חדר "יותר טוב" מ-self (כפי שהוגדר למעלה במתודה better_than), אז רמת שביעות הרצון של האורחים שעברו ל-other משתפרת, ולכן כעת $other.satisfaction = \min(5.0, self.satisfaction + 1.0)$. <u>אחרת,</u> רמת שביעות הרצון ב-other הופכת לזו שב-self בעת ביצוע ההעברה. ג. לבסוף, מחיקת איברי רשימת האורחים של self, כך שהיא הופכת לריקה. • ניתן להניח ש-other הוא אובייקט תקין מסוג Room. • ניתן להניח ש-self ו-other מייצגים חדרים שונים.

<u>הערה:</u> ניתן בפייתון להעלות חריגים (Exceptions) מיוחדים שאותם הגדרנו בעצמנו לפי הצרכים שלנו בתוכנית מסויימת. דוגמה לכך היא **סוג החריג RoomError, שמוגדר בקובץ התרגיל, וניתן להעלות אותו עם raise** כפי שעשינו עד כה עם כל חריג סטנדרטי.

<u>דוגמת הרצה (וודאו שהנכם מבינים היטב את מהלכה):</u>

```
>>> r1 = Room(m, 2, 23, ["Dana", "Ron"], 5, 2)
>>> r_better = Room(m, 6, 57, [], 4, 3)
>>> r_better.better_than(r1)
True
>>> r_better.check_in(["Amir"])
>>> r_better.clean()
>>> r_better.clean_level
7
>>> r1.check_in(["Avi", "Hadar"])
Traceback (most recent call last):
…
RoomError: Cannot check-in new guests to an occupied room
>>> r1.is_occupied()
```

```
True
>>> r1.check_out() ## note: None is returned, and so nothing is printed
>>> r1.is_occupied()
False
>>> r_better.move_to(r1)
>>> r1.satisfaction
1.0
>>> r1.guests
['amir']
>>> r1.move_to(r_better)
>>> r1.is_occupied()
False
>>> r_better.satisfaction
2.0
>>> r_better.guests
['amir']
```

שאלה 3

כעת נממש את המחלקה Hotel המייצגת בית מלון.

א. ממשו את הבנאי __init__(self, name, rooms) המקבל מחרוזת name המציינת את שם המלון, ורשימת חדרים rooms, שאין לבדוק את תקינותם.

הערות:

- ניתן להניח ש-name הינו מחרוזת המייצגת שם מלון חוקי, שמכילה רווחים, ספרות ואותיות אנגליות בלבד (ב- uppercase ו\או ב-lowercase).
- ניתן להניח שהרשימה rooms לא ריקה, כאשר כל איבריה הינם חדרים תקינים (מסוג Room) ושונים זה מזה (כלומר, אין אובייקט חדר המופיע פעמיים, ואין אובייקטים שונים עם אותו מספר חדר וגם אותו מספר קומה). ניתן להניח ששמות האורחים שונים זה מזה גם באותו החדר וגם בחדרים השונים.
- הרשימה rooms יכולה להכיל חדרים תפוסים ו\או פנויים.
- ניתן לשמור את החדרים כשדה של אובייקט המלון תוך שימוש בכל מבנה נתונים בפייתון שהנכם רואים לנכון. בפרט, אין הכרח להשתמש ברשימה במימוש הפנימי.
- ניתן להוסיף שדות ו/או מתודות נוספים שיכולים לסייע לכם במימוש המחלקה.

ב. ממשו את המתודה __repr__(self) אשר מחזירה מחרוזת שמתארת את אובייקט המלון על פי הפורמט הבא:

```
"<self.name><רווח בודד>hotel has:\n
<number of room objects><רווח בודד>rooms\n
<number of occupied rooms><רווח בודד>occupied rooms"
```

דוגמת הרצה:

```
>>> h = Hotel("Best",[Room(m, 15, 140, [], 5, 1), Room(m, 1, 2, ["Liat"],
7, 4)])
>>> h
Best hotel has:
2 rooms
1 occupied rooms
```

הערה: בכתיבת הקוד של __repr__, חישוב מספר החדרים יכול להתבצע בכל אופן שהנכם רואים לנכון. בפרט, ניתן להשתמש במתודות עזר.

ג. על אובייקט מלון לתמוך במתודות הבאות:

שם וחתימה	תיאור
check_in(self, guests, rank)	מנסה לבצע צ'ק-אין לרשימת שמות האורחים guests (רשימת מחרוזות) לחדר אחד (כלשהו) מחדרי המלון, שדרגתו היא rank (מספר שלם). • במידה ונמצא חדר פנוי ומתאים בדרגתו, המתודה תבצע ל-guests צ'ק-אין אליו, ותחזיר את (אובייקט) החדר שנמצא, ואחרת – None. • ניתן להניח ששמות האורחים ב-guests לא מתנגשים עם שמות אורחים אחרים שכבר שוהים במלון. • ניתן להניח ש-guests היא רשימת מחרוזות לא ריקה עם שמות חוקיים (כלומר, שכוללים רק אותיות אנגליות ורווחים), ואותיות שיכולות להיות בפורמט case גדול או קטן.
check_out(self, guest)	מנסה לבצע צ'ק–אאוט לאורח בשם guest (מחרוזת) יחד עם האורחים הנוספים השוהים עמו בחדר (אם יש כאלה). • במידה ונמצא החדר בו הוא שוהה, יש לבצע את הצ'ק-אאוט בהצלחה, ולהחזיר את אובייקט החדר בו שהה האורח. • אחרת – לא ניתן לבצע את הצ'ק אאוט, ויש להחזיר None. • בחיפוש החדר בו guest שוהה יש להתעלם מפורמט ה-case. למשל, נתייחס ל-UZI כ-uzi.
upgrade(self, guest)	מנסה לבצע "שדרוג" חדר לאורח בשם guest (מחרוזת), במידה ו-guest שוהה בחדר בבית המלון, ויש חדר פנוי שניתן "לשדרג" אליו. • פעולת ה"שדרוג" כוללת את העברת האורח יחד עם האורחים הנוספים השוהים עמו בחדר (אם יש כאלה) לחדר אחר במלון שהינו פנוי ו"טוב יותר" (כפי שהוגדר בשאלה 1). • במידה וה"שדרוג" מתבצע בהצלחה, יש להחזיר את (אובייקט) החדר שאליו הועברו האורחים. • אחרת, אם האורח לא שוהה במלון או פעולת ה"שדרוג" לא ניתנת לביצוע, יש להחזיר None. • בחיפוש החדר בו guest שוהה יש להתעלם מה-case. • במידה וישנם מספר חדרים הניתנים לשדרוג יש לשדרג לאחד מהם

הערות:

- על כל המתודות תמיד לסיים לרוץ ללא העלאת חריגים כלשהם מסוג RoomError

- ניתן להניח את תקינות (סוג וערך) הארגומנטים בכל המתודות. בפרט, ניתן להניח שכל מחרוזת בקלט מייצגת שם תקין של אורח ב-case גדול ו/או קטן.

<u>דוגמת הרצה:</u>

- קוד הרצה לדוגמה מצורף לשלד התרגיל בפונקציה **test_hotel**.
- הקובץ **test_hotel_output.txt (המצורף לתרגיל)** כולל את ההדפסות שנוצרות במהלך ריצת הפונקציה הנ"ל, ונועד לאפשר לכם לבדוק שהדפסות המימוש שלכם זהות.
- שימו לב, ישנן פקודות להן יתכן יותר מערך אחד תקין (למשל ישנן מספר אפשרויות לשדרוג החדר של liat), במקרה כזה יתכן שיתקבל פלט השונה מזה שבדוגמה (תלוי מימוש).

בהצלחה!