

שאלה 2 – OOP (38 נקודות)

חברת ITube מעוניינת לשפר את חווית הצפייה בוideo (video) בקרב משתמשים ועל כן היא החליטה למשם מחדש את מוצר הדגל שלה בשפת Python.

סעיף א' (10 נקודות):

משמעותה של מחלקת Clip אשר מייצגת סרטון וידאו. סרטון וידאו מאופיין על ידי:

- name - שם הסרטון (משתנה מטיפוס string).
- year - שנת ההפקה של הסרטון (משתנה טיפוס integer).
- is_played - משתנה בוליאני המציין את מצב הניגון של הסרטון כרגע: True עבור "מנוגן" ו False עבור "לא-מנוגן".

המחלקה תכלול את המתודות הבאות:

1. המתודה `__init__(self, name, year)` מתחילה מופיע חדש של המחלקה כך שיכלול שדה בשם name המכיל את שם הסרטון, שדה בשם year המכיל את שנת ההפקה, ושדה בשם is_played המכיל את מצב הניגון של הסרטון (בתחילה, הסרטון מוגדר במצב "לא-מנוגן"). אין צורך לבדוק תקינות קלט.
 2. המתודה `__repr__(self)` מחזירה ייצוג של הסרטון מהצורה "name:year" למשל: "Gangnam Style:2012". זכרו: שנת ההפקה שמורה בתור integer.
 3. המתודה `play(self)` משנה את מצב הניגון של הסרטון למצב "מנוגן".
 4. המתודה `stop(self)` משנה את מצב הניגון של הסרטון למצב "לא-מנוגן".
- אין צורך לבדוק במתודות play ו stop את המצב הנוכחי, אלא רק להעביר למצב החדש.

דוגמא לשימוש במחלקה:

```
>>> c1 = Clip("Gangnam Style", 2012)
>>> print c1
Gangnam Style:2012
>>> print c1.is_played
False
>>> c1.play()
>>> print c1.is_played
True
```

```
class Clip:  
    def __init__(self, name, year):  
        self.name = name  
        self.year = year  
        self.is_played = False  
  
    def __repr__(self):  
        return self.name + ":" + str(self.year)  
  
    def play(self):  
        self.is_played = True  
  
    def stop(self):  
        self.is_played = False
```

כל הזכויות שמורות ©
מכללי לפניו כאמור לעיל, אין להעתיק, לצלם, להקליט, לשדר, לאחסן מגיר מידע, בכל דרך שהיא, בין מכנית ובין אלקטרוני או בכל דרך אחרת כל חלק שהוא מטופש הבדיקה.

סעיף ב' (8 נקודות):

משמעותו מחלוקת נוספת בשם ArtClip המייצגת סרטון אמנות. לחלוקת זו יש את כל השודות והמתודות שלחלוקת Clip, אך עם השינויים והתוספות הבאים:

1. בניית החלוקת (Clip) מקבלת את שם הסרטון, את שנת התפקידתו (מספר שלים), ושומר אותו בשדה פנימי בשם

.time

2. מחרוזת ייצוג האובייקט המוחזרת על ידי מתודת `__repr__` תהיה זהה לייצוגו של אובייקט מסוג Clip, אך תכיל בסופה גם את זמן הסרטון בדקות בתוך סוגרים עגולים לאחר רווח בודד (ראו דוגמא בהמשך).

3. החלוקת מכיל גם את המתודה `is_valid_for_euro(self)`, המחזירה True אם הסרטון עומד בקריטריונים להשתתפות בתחרויות סרטוני האמנויות האירופית, ו-False אחרת. הקритריון להשתתפות הוא שאורך הסרטון אינו עולה על שלוש דקות.

משמעותו מחלוקת תוך שימוש שכפול קוד ככל האפשר ובהנחה שיש בידיכם כבר מימוש של החלוקת Clip אותה הגדרתם בסעיף א'.

דוגמא לשימוש בחלוקת:

```
>>> c2 = ArtClip("Gangnam Style", 2012, 3)
>>> print c2
Gangnam Style:2012 (3)
>>> print c2.is_valid_for_euro()
True
>>> c3 = ArtClip("Moon", 2001, 4)
>>> print c3.is_valid_for_euro()
False
```

```
class ArtClip(Clip):  
  
    def __init__(self, name, year, time):  
        Clip.__init__(self, name, year)  
        self.time = time  
  
    def __repr__(self):  
        return Clip.__repr__(self)+" ("+str(self.time)+")"  
  
    def is_valid_for_euro(self):  
        return self.time <= 3
```

סעיף ג' (10 נקודות)

- עתה ממשו מחלוקת בשם Player אשר מייצגת נון סרטונים. הננו מאופיין על ידי :
- clips - רשימה של סרטונים (משתנה מטיפוס list המכיל אובייקטים מסווג Clip או .(ArtClip).
 - current – מספר המציין את האינדקס ברשימה clips של הסרטון שמנוגן, או 1- אם אף סרטון לא מנוגן (משתנה מטיפוס int).

המחלקה תכלול את המתודות הבאות :

1. המתודה `__init__(self, clips)` מתחילה מופיע של המחלוקת עם משתנה פנימי בשם `clips` המכיל את רשימת הסרטונים. על המתודה להפסיק את הניגון של כל הסרטונים שברשימה, ולהציב בשדה `current` את הערך 1.
2. המתודה `__next__(self)` מקדמת את הנגן הסרטון הבא ברשימה הסרטונים. על המתודה להפסיק את ניגון הסרטון הנוכחי ולהתחל לנגן את הסרטון הבא. אם אף סרטון לא מנוגן, יש להתחיל לנגן את הסרטון הראשון. אם הסרטון האחרון מנוגן, אז יש להפסיק אותו ולהתחל לנגן את הסרטון הראשון. שימוש לב שיש לעדכן גם את השדה `current` כך שיצביע על הסרטון שכרגע מנוגן.
במידה ורשימת הסרטונים ריקה והמתודה הופעלה – יש להעלות חריג מסוג `.ValueError`.

```
class Player:  
    def __init__(self, clips):  
        self.clips = clips  
        for clip in clips:  
            clip.stop()  
        self.current = -1  
  
    def next(self):  
        if len(self.clips) == 0:  
            raise ValueError  
        if self.current > -1:  
            self.clips[self.current].stop()  
        self.current = (self.current + 1) % len(self.clips)  
        self.clips[self.current].play()
```

סעיף ד' (10 נקודות):

בחברת ITube החלטו להוסיף לנגן תמייה בערבוב (shuffle) של רשיימת הסרטונים. הוסיףו מימוש של המתודה `shuffle(self)` למחלקה `Player`, אשר מעורבת את רשיימת הסרטונים בנגן, ויוצרת סידור מקרי חדש. ניתן להניח שאין סרטון המנוגן בעת הקריאה למетодה.

הנחיות:

- על המתודה להשתמש בפונקציה `random.randint(a,b)` המגרילה ומחזירה מספר רנדומלי שלם בתחום `[a,b]` (כולל נקודות הקצה).
- על הפונקציה לשנות את רשיימת הסרטונים הקיימת של הנגן, כך שתהיה סידור מחדש של הרשימה המקורי (אין לחזור על אותו סרטון פעמיים). המתודה אינה מחזירה דבר.
- על הסידור החדש להיות **קרי**, כלומר אין לבצע טרנספורמציה קבועה על סדר הסרטונים (היפוך, הזזה צדנית, וכו').
- איןכם מוגבלים במספר הבדיקות או הגרלות שהמתודה מבצעת.

```
class Player:

    ...

    def shuffle(self):
        temp = []
        chosen_indices = []
        for index in range(len(self.clips)):
            while True:
                randomindex = random.randint(0,len(self.clips)-1)
                if (randomindex not in chosen_indices):
                    temp.append(self.clips[randomindex])
                    chosen_indices.append(randomindex)
                    break
        self.clips = temp
```