מבוא לתכנות 52304

תרגיל 4 - לולאות ורשימות

להגשה בתאריך 15.12.2019 בשעה 23:59

בתרגיל זה נתרגל שימוש בלולאות ומשתנים. דגשים לתרגיל:

- כתבו את כל הקוד שלכם בתוך הקובץ ex4.py.
- **חתימות הפונקציות** (שם הפונקציה והפרמטרים שלה) במימוש שלכם צריכות להיות זהות במדויק לחתימות המתוארת במשימות. ניתן לשנות את חתימות הפונקציות על ידי הוספת ערכים דיפולטים (ערכי ברירת מחדל) לפרמטרים של פונקציות (במקומות בהם השימוש מתאים).
- לנוחותכם, מסופק **קובץ שלד** ex4.py הכולל את כל חתימות הפונקציות שיש לממש.
- ניתן להוסיף לקובץ המוגש **פונקציות נוספות** במידת הצורך וניתן להשתמש בשאלות מתקדמות בפונקציות שמימשתם בשאלות קודמות.
- **סגנון**: הקפידו על תיעוד נאות ובחרו שמות משתנים משמעותיים. הקפידו להשתמש בקבועים (שמות משתנים באותיות גדולות) על פי הצורך.
- שימוש ב**כלים שלא נלמדו בקורס** – אין להשתמש במודולים פרט לאלו שנכתבו בשאלה. בפרט, **אין להשתמש ב**מודולים של numpy וב – itertools ואין להשתמש בפונקציית count. במקרה בו יש ספק - יש לפנות לצוות הקורס.
- **רשימה ריקה** – רשימה שאינה מכילה איברים: []. היא עדיין נחשבת רשימה ומספר האיברים בה הוא 0.
- שימו לב מתי יש לקבל **קלט מהמשתמש** בעזרת הפונקציה input (שאלה 7 בלבד) ומתי הקלט מתקבל כארגומנט בעת קריאה לפונקציה. כאשר הפונקציה לא מקבלת קלט מהמשתמש בעזרת פונקציית input, מומלץ לכתוב בעצמכם קטעי קוד אשר קוראים לפונקציות עם ארגומנטים מתאימים כדי לבדוק שהן עובדות. לנוחיותכם מצורף קטע קוד כזה בשם run_dobble.py עם דוגמאות בו תוכלו להשתמש ולשנות אותו כרצונכם. (אין לכלול את קטעי קוד הבדיקה בהגשה הסופית).
- בכל שאלה מפורט **מה ניתן להניח על הקלט** - אין צורך לבצע בדיקות תקינות נוספות מעבר למפורט (כלומר, ניתן להניח כי הקלט תקין).
- בכל הסעיפים, כאשר **פונקציה צריכה להחזיר ערך**, הכוונה היא לשימוש במילה השמורה **return**, בפרט הכוונה אינה להדפיס למסך בעזרת הפונקציה print. שימו לב, **אין להדפיס למסך** כל הודעה מלבד ההודעות הנדרשות במפורש.
- **ניקוד:** בתרגיל זה 7 שאלות ובנוסף שאלת בונוס. הניקוד לכל שאלה מופיע לידה.

התרגיל עוסק במימוש של משחק הקלפים הפופולרי dobble. במשחק זה יש חבילה של n קלפים, כאשר על כל קלף מודפסים k סמלים שונים. החבילה מתוכננת כך שעבור כל זוג קלפים יש **בדיוק** סמל אחד משותף לשני הקלפים. לדוגמא עבור 2 הקלפים שבתמונה, הסמל המשותף הוא עכביש 'spider'. מטרת המשחק בכל סיבוב היא למצוא את הסמל המשותף במהרה.

במשחק המקורי מספר הקלפים הוא n=55 ומספר הסמלים בכל קלף הוא k=8.אך אנו נממש משחק כללי יותר כאשר n,k יהיו פרמטרים. במשחק המקורי יש שני שחקנים, כשהשחקן המהיר יותר הוא הזוכה בכל סיבוב. אנו נממש גרסא של המשחק לשחקן אחד, כשהמטרה היא למצוא את הסמל המשותף כמה שיותר מהר.

כדי לייצג את חבילת הקלפים בפייתון נשתמש ברשימה של רשימות. מספר האיברים ברשימה יהיה n, כאשר כל איבר ברשימה מייצג קלף. כל קלף ייוצג על ידי רשימה באורך k, המייצגת את הסמלים בקלף זה. לדוגמא, חבילה המכילה את שני הקלפים שבתמונה תיוצג כך:

deck = [[‘dolphin’, ‘bomb’, ‘spider’, ‘pencil’, ‘treble clef’, ‘scarecrow’, ‘sun’, ‘stop’],

[‘apple’, ‘snowflake’, ‘iglu’, ‘moon’, ‘maple leaf’, ‘scissors’, ‘spider’, ‘exclamation point’]]

הערה: על אף שהפונקציות נדרשות להשתמש ברשימות עבור הקלפים, בחלק מהשאלות מומלץ להמיר את המשתנים לטיפוס set כדי לבצע את המשימה. בסוף יש להחזיר לטיפוס list את הפלט כנדרש בפונקציות.

הנחות קלט: אלא אם נאמר אחרת, בכל השאלות ניתן להניח שחפיסת הקלפים מכילה קלפים **שונים**, וכן שבכל קלף הסמלים הם **שונים**. אין להניח שהחפיסה תקינה אם לא נאמר זאת.

1. [10 נק'] כתבו פונקציה בשם cards_intersect הבודקת עבור 2 קלפים מהו החיתוך שלהם. כלומר אילו סמלים מופיעים בשני הקלפים. הפונקציה מקבלת שתי רשימות card1, card2 ומחזירה רשימה חדשה in_both בה מופיעים הסמלים המשותפים לשני הקלפים. לדוגמא, עבור:

card1 = [‘tortoise’, ‘dog’, ‘zebra’]

card2 = [‘zebra’, ‘tortoise’, ‘cheese’]

הקריאה לפונקציה:

cards_intersect(card1, card2)

תחזיר את הרשימה:

[‘tortoise’, ‘zebra’]

שימו לב שהפונקציה צריכה לעבוד בצורה נכונה גם עבור קלפים שאינם חלק מחבילה חוקית - כלומר כאלו בעלי אורך שונה, או עם מספר סמלים משותף השונה מאחד.

אין חשיבות לסדר האיברים ברשימה המוחזרת - כלומר ניתן להחזירם בסדר כלשהו, אך כל סמל חייב להופיע בדיוק פעם אחת.

2. [12 נק'] כתבו פונקציה בשם remove_card המסירה קלף מהחפיסה. הפונקציה מקבלת שני פרמטרים:
deck - רשימה של רשימות המייצגת את החפיסה
card - רשימה המייצגת את הקלף אותו יש להסיר
על הפונקציה לבדוק שהקלף שהתווסף נמצא בחפיסה, ואם כן אז להסיר אותו. **שימו לב:** אם מופיע בחפיסה קלף עם אותם סמלים אך בסדר שונה מהקלף אותו התבקשתם להסיר, אז עדיין יש להסירו!
אם הקלף לא נמצא, יש להדפיס הודעת שגיאה:

Error! card is not in the deck

לדוגמא, עבור החבילה הבאה של n=4 קלפים עם k=3 סמלים:

deck = [[‘dolphin’, ‘bomb’, ‘spider’], [‘eye’, ‘bomb’, ‘fire’],
[‘spider’, ‘fire’, ‘lock’], [‘bomb’, ‘lock’, ‘tree’]]

אחרי הקריאה לפונקציה:

remove_card(deck, [‘spider’, ‘fire’, ‘lock’])

החבילה תיראה כך:

[[‘dolphin’, ‘bomb’, ‘spider’], [‘eye’, ‘bomb’, ‘fire’], [‘bomb’, ‘lock’, ‘tree’]]

- הפונקציה מחזירה ערך True אם הצליחה להסיר את הקלף וערך False אם לא הצליחה

3. [12 נק'] כתבו פונקציה בשם add_card המוסיפה קלף חדש לחפיסה. הפונקציה מקבלת שני פרמטרים:
deck - רשימה של רשימות המייצגת את החפיסה
card - רשימה המייצגת את הקלף הנוסף
על הפונקציה לבדוק שהקלף שהתווסף הוא חוקי, כלומר רשימה באורך הקיים בחפיסה. כמו כן צריך לבדוק שלקלף החדש יש בדיוק סמל אחד משותף עם כל אחד מהקלפים הקיימים בחפיסה. אם תנאי זה מתקיים, יש להוסיף את הקלף לחפיסה, כלומר לעדכן את הרשימה deck כל ש-card יהיה אחד מאיבריה. אם התנאי לא מתקיים, יש להדפיס הודעת שגיאה בהתאם למקרה:

Error! number of matches for new card is not one

או:

Error! Card is of wrong length

לדוגמא, עבור החבילה deck מהשאלה הקודמת, ועבור:

card1 = [‘spider’, ‘eye’, ‘tree’]

card2 = [‘candle’, ‘carrot’, ‘hammer’]

לאחר הקריאה:

add_card(deck, card1)

החבילה deck תראה כך:

[[‘dolphin’, ‘bomb’, ‘spider’], [‘eye’, ‘bomb’, ‘fire’], [‘spider’, ‘fire’, ‘lock’],
[‘bomb’, ‘lock’, ‘tree’], [‘spider’, ‘eye’, ‘tree’]]

לאחר הקריאה:

add_card(deck, card2)

נקבל הודעת שגיאה:

Error! number of matches for new card is not one

- הפונקציה מחזירה ערך True אם הצליחה להוסיף את הקלף וערך False אם לא הצליחה

4. [16 נק'] כתבו פונקציה הבודקת שהחבילה תקינה. הפונקציה בשם is_valid מקבלת פרמטר אחד deck שהיא רשימה של רשימות המייצגת את החפיסה. הפונקציה מחזירה ערך בוליאני כאשר True יוחזר אם החפיסה תקינה ו-False אם היא אינה תקינה. חבילה היא תקינה אם היא מקיימת את התנאים הבאים:

- כל הקלפים מכילים את אותו מספר סמלים, כאשר כל סמל הוא מחרוזת
- עבור כל זוג קלפים יש בדיוק סמל אחד משותף

5. [10 נק'] כתבו פונקציה בשם draw_random_cards המקבלת כקלט רשימה של רשימות המייצגת את החבילה, בוחרת שני קלפים **שונים** באקראי מתוך החפיסה, ומחזירה אותם. לדוגמא עבור הרשימה:

deck = [['key', 'tree', 'dog'], ['sun', 'stop', 'dog'], ['ghost', 'tree', 'sun'], ['sun', 'key', 'lock']]

והפקודה:

draw_random_cards(deck)

אם נבחרו הקלפים מספר 0 ו-3, הפונקציה תחזיר:

[['key', 'tree', 'dog'], ['sun', 'key', 'lock']]

- **הנחות על הקלט:** הרשימה deck היא באורך של 2 או יותר וכל האיברים בה הם רשימות באורך שווה המכילות רק מחרוזות מטיפוס string.
- כדי להגריל את 2 הקלפים יש להשתמש בפונקציה random.sample מהמודול random אותו יש לייבא.
- יש להחזיר רשימה של רשימות בפורמט זהה לרשימת הקלט deck, אך המכילה רק 2 קלפים.

6. [16 נק'] כתבו פונקציה בשם print_symbols_counts המקבלת רשימה deck המייצגת את חבילת הקלפים, ומדפיסה את כל הסמלים המופיעים באיזשהו קלף, וכן עבור כל סמל את מספר הפעמים הכולל שהוא הופיע בחפיסה, כאשר בכל שורה מופיע הסמל ומספר הפעמים, מופרדים ברווח. לדוגמא, עבור החבילה:

deck = [['dolphin', 'bomb', 'spider'], ['eye', 'bomb', 'fire'],

['spider', 'fire', 'lock'], ['bomb', 'lock', 'tree']]

אחרי הקריאה:

Symbol, counts = print_symbols_counts(deck)

יודפס למסך (בסדר כלשהו):

dolphin 1
bomb 3
spider 2
eye 1
fire 2
lock 2
tree 1

7. [24 נק'] כתבו פונקציה בשם play_dobble המאפשרת לשחקן לשחק במשחק באופן אינטראקטיבי.

- הפונקציה מקבלת פרמטר deck המייצג חבילת קלפים ולא מחזירה דבר. מותר להניח שהחבילה תקינה.
- לאחר קריאה לפונקציה, הפונקציה תדפיס למסך את ההודעה:

Select operation: (P)lay, (A)dd card, (R)emove card, or (C)ount

ותמתין לקלט מהמשתמש של אות אחת. הפונקציה תבצע את הפעולות הבאות לפי הקלט מהמשתמש:

- עבור קלט A, הפונקציה תמתין להכנסת הקלף אותו יש להוסיף, עם סמלים מופרדים בפסיקים, ותנסה להוסיפו על פי הכללים של שאלה 3, כולל הודעת השגיאה המתאימה אם לא ניתן להוסיף.
- עבור קלט R, הפונקציה תמתין להכנסת הקלף אותו יש להסיר, עם סמלים מופרדים בפסיקים, ותנסה להסירו על פי הכללים של שאלה 2, כולל הודעת השגיאה המתאימה אם לא ניתן להסיר.
- עבור קלט C, הפונקציה תדפיס את כל הסמלים בחפיסה כולל מספרם באמצעות שימוש בפונקציה משאלה 6.
- עבור קלט P, הפונקציה תתחיל במשחק. בכל סיבוב יוצגו 2 קלפים אקראיים ולאחר תום הסיבוב הם יוסרו מהחפיסה, עד שיישארו בחפיסה פחות מ-2 קלפים ואז המשחק יסתיים. בכל סיבוב:
  - היא תדפיס למסך את ההודעה (שימו לב, לאחר הנקודתיים בסוף השורה רווח יחיד):

Identify joint symbol:

ותדפיס שני קלפים אקראיים מהחפיסה, כל קלף בשורה נפרדת, עם סמלים מופרדים ע"י פסיק.

  - הפונקציה תמתין לקלט של מחרוזת מהמשתמש ותבדוק אם היא אכן מתאימה לסמל המשותף ותדפיס בהתאם עבור מחרוזת נכונה את הזמן שלקח למשתמש למצוא את הסמל, באופן הבא:

Very Nice! Found the correct card in 2.47 sec.

ועבור מחרוזת לא נכונה:

Wrong!

אפשר להניח קלטים תקינים: מחרוזת באורך שורה אחת.

כדי למדוד את הזמן שלקח למשתמש למצוא את הקלף השתמשו בפקודה time.time() מתוך המודול time, בתחילת ובסיום הפעולה. עגלו את הזמנים ל-2 ספרות אחרי הנקודה בעזרת הפונקציה round.

בסוף המשחק, כשנשארו פחות מ-2 קלפים בחבילה, הפונקציה תדפיס את מספר הזיהויים הנכונים ומספר הזיהויים הלא נכונים, וכן את הזמן הממוצע לזיהוי נכון (או 0 אם לא היה אף זיהוי נכון) לדוגמא:

Finished Game. Correct: 2 Wrong: 1 Average time: 6.68

עבור אופציה זו (P), לאחר היציאה מהפונקציה בתום המשחק, על החפיסה **להישאר כפי שהיתה** לפני הקריאה לפונקציה, **ולא** לקטון כתוצאה מהסרת זוגות הקלפים

- דוגמא להרצה:

```
>>> play_dobble(deck)
Select operation: (P)lay, (A)dd card, (R)emove card, or (C)ount
>? P
Identify joint symbol:
sun, spider, exclamation point, scarecrow, moon
zebra, eye, treble clef, heart, moon>? moon
Very nice! Found the correct card in 3.34 sec.
Identify joint symbol:
zebra, spider, snowman, scissors, snowflake
tree, iglu, bomb, eye, spider>? spider
Very nice! Found the correct card in 7.45 sec.
Finished Game. Correct: 2 Wrong: 0 Average time: 5.40 sec.
```

8. **שאלת בונוס:** כתבו פונקציה בשם create_deck המקבלת בתור קלט רשימת סמלים symbols וכן את מספר הסמלים בכל קלף k. על הפונקציה להחזיר חפיסת קלפים תקנית באורך n גדול ככל האפשר, כאשר כל קלף מכיל k סמלים מתוך הרשימה symbols.

לדוגמא, עבור הרשימה

```
symbols = ['stop', 'bomb', 'moon']
```

החפיסה הארוכה ביותר של קלפים עם שני סמלים היא באורך 3. הקריאה:

```
create_deck(symbols, 2)
```

תחזיר:

```
[['stop', 'bomb'], ['stop', 'moon'], ['moon', 'bomb']]
```

מספר נקודות הבונוס שתקבלו עבור חפיסה **תקנית** שתוחזר כפלט יהיה גודל החפיסה פחות 5,
עבור קלט של רשימת symbols באורך 20 וכאשר בכל קלף יש k=5 סמלים

בדיקה עצמית של הקבצים: ראו הוראות במסמך "נוהל הרצת הטסטים" באתר

הוראות הגשה

עליכם להגיש את הקובץ ex4.zip (בלבד) בקישור ההגשה של תרגיל 4 דרך אתר הקורס על ידי לחיצה על "Upload file".

ex4.zip צריך לכלול את הקבצים:

1. ex4.py
2. README (כמפורט בנהלי הקורס)

הנחיות כלליות בנוגע להגשה

- הנכם רשאים להגיש תרגילים דרך מערכת ההגשות באתר הקורס מספר רב של פעמים. ההגשה האחרונה בלבד היא זו שקובעת ושתיבדק.
- לאחר הגשת התרגיל, ניתן ומומלץ להוריד את התרגיל המוגש ולוודא כי הקבצים המוגשים הם אלו שהתכוונתם להגיש וכי הקוד עובד על פי ציפיותיכם.
- ניתן לאחר בהגשת התרגיל עד שלושה ימים, אך זה יהיה כרוך בניכוי ציון, כפי שמפורט באתר הקורס ובמסמך נהלי הקורס.
- קראו היטב את קובץ נהלי הקורס לגבי הנחיות נוספות להגשת התרגילים.
- שימו לב - יש להגיש את התרגילים בזמן!

בהצלחה!