תרגיל בית 10 (ואחרון!)

numpy, pandas, image processing

<u>הנחיות כלליות:</u>

- קראו היטב את השאלות והקפידו שהתכניות שלכם פועלות בהתאם לנדרש.
- את התרגיל יש לפתור לבד!
- אין לשנות את שמות הפונקציות והמשתנים שכבר מופיעים בקובץ השלד של התרגיל.
- אין למחוק את ההערות שמופיעות בשלד.
- **יש לבצע בדיקה עצמית אך ורק בסוף קובץ השלד בתוך התנאי הבא:**
  if __name__ == "__main__":
  בפרט אין להשאיר הדפסות/קריאות לפונקציות בין הפונקציות.
- **אין להשאיר בקוד נתיבים לקבצים מקומיים.**
- הקפידו על כללי ההגשה המפורסמים באתר. בפרט, יש להגיש את כל השאלות יחד בקובץ ex10_012345678.py המצורף לתרגיל, לאחר החלפת הספרות 012345678 במספר ת.ז שלכם, כל 9 הספרות כולל ספרת הביקורת.
- <u>אופן ביצוע התרגיל:</u> בתרגיל זה עליכם לממש את הפונקציות הנתונות. ניתן להוסיף פונקציות עזר.
- מועד אחרון להגשה: כמפורסם באתר.
- היות ובדיקת התרגילים עשויה להיות אוטומטית, יש להקפיד על פלטים מדויקים על פי הדוגמאות (עד לרמת הרווח).
- <u>בדיקה עצמית:</u> 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 וכי התוכנית אינה קורסת).
- בכל השאלות ניתן להניח את תקינות הקלט על פי המפורט בשאלה.

שאלה 1 – numpy

בתרגיל זה נממש קוד להסתרה של הודעות טקסט קצרות בתוך תמונות. שיטה זו להעברה חשאית של הודעות בתוך תמונות תמימות ידועה בשם - סטגנוגרפיה (steganography).

https://en.wikipedia.org/wiki/Steganography

כמו שראינו בכיתה, ניתן להשתמש במערכים דו מימדיים של Numpy לייצוג תמונות בגווני אפור. כל איבר במערך הדו-מימדי הוא מספר שלם המייצג את גוון האפור של פיקסל אחד בתמונה. בתרגיל זה הטיפוס של כל איבר במערך יהיה uint8 (מספר שלם שמקודד בבייט בודד), המסוגל לייצג ערכים בטווח 0-255 ועל כן ניתן לייצג איתו 256 גווני אפור, אך גם אפשר לתת לו משמעות של תו במחרוזת כלשהי. אנחנו ננצל את התכונה הזאת כדי להסתיר הודעות בתוך תמונות.

עבור תמונה נתונה ומחרוזת קצרה כלשהי, נכתוב קוד שיחזיר תמונה חדשה בה מקודדת ההודעה באופן שישפיע כמה שפחות על מראה התמונה. כדי שיהיה ניתן לחלץ את ההודעה מהתמונה, נשמור במערך המייצג את התמונה גם נתונים נוספים: המיקום בתמונה בה שולבה ההודעה, ואורך המחרוזת.

כיצד נשלב הודעה (רצף של מספרים המייצגים תווים) בתוך תמונה בגווני אפור (המיוצגת על ידי מערך דו מימדי של מספרים) בצורה הסודית ביותר? נתחיל בתיאור כללי של שלבי האלגוריתם:

1. האלגוריתם אותו נממש יתחיל בהמרת טקסט ההודעה לרצף של מספרים תוך שימוש בפונקציית **ord()**, המחזירה לכל תו בודד את ערך ה-ASCII שלו.
לדוגמא:

```
>>>print(ord('H'), ord('e'), ord('l'), ord('l'), ord('o'))
```

72 101 108 108 111

בקובץ התרגיל כבר נתונות לכם שתי פונקציות עזר שמבצעות המרה ממחרוזת טקסט למערך Numpy של מספרים מסוג uint8 המכיל את ערכי ה-ASCII של כל תו ולהיפך. שמות הפונקציות: **np_array_to_ascii** ו-**ascii_to_np_array**. דוגמת שימוש:

```
>>> x = ascii_to_np_array("hello")
>>> x
array([104, 101, 108, 108, 111], dtype=uint8)
>>> np_array_to_ascii(x)
'hello'
```

2. האלגוריתם ימשיך במציאת מיקום (זוג אינדקסים- שורה ועמודה) במערך התמונה שבו מתחיל רצף של מספרים שהינו בעל רמת הדימיון הגבוהה ביותר לרצף המספרים המקודדים את הודעת הטקסט. החלפת רצף מספרים זה במערך התמונה עם רצף המספרים המקודדים את ההודעה ישפיע בצורה מינימלית על מראה התמונה. מיקום תחילת ההודעה אינו יכול להכיל אינדקס שערכו גדול מ-255 (ראו הערות). כמו כן, על ההודעה להיות מוכלת בשורה בודדת של התמונה. משמע, אם אורך ההודעה הוא k, אזי לא ניתן למקם את נקודת ההתחלה שלה k - 1 ומטה פיקסלים מהקצה הימני של התמונה. ניתן להניח כי הקלט שתקבלו יעמוד בתנאים אלו.

3. ליצירת תמונה בה משולבת ההודעה, נבצע את הצעדים הבאים:
3.1. נעתיק את מערך הקלט שמייצג את התמונה למערך חדש בו ניצור את תמונת הפלט.
3.2. נמצא את המיקום הכי טוב בתמונה לשתילת ההודעה.
3.3. נעתיק את רצף המספרים המקודדים את ההודעה אל המיקום שמצאנו בסעיף הקודם.
3.4. נשמור מידע נוסף בפיקסלים בפינה הימנית התחתונה במערך תמונת הפלט שיאפשר חילוף של ההודעה בעתיד:

פיקסל [-1, -3] – אינדקס השורה אליה הועתקה ההודעה
פיקסל [-1, -2] – אינדקס העמודה אליה הועתקה ההודעה
פיקסל [-1, -1] –אורך ההודעה

שימו לב כי המשמעות של אינדקס שלילי במערך Numpy זהה למשמעות שלו ברשימה פייתונית רגילה. משמע, הפיקסל במיקום ה [-1, -1] הינו הפיקסל הימני ביותר (כלומר בעמודה האחרונה) בתחתית התמונה (כלומר בשורה האחרונה).

הערות

1. התמונה אינה בהכרח ריבועית.
2. ניתן להניח שאורך ההודעה קטן ממספר העמודות בתמונה.
3. ניתן להניח שההודעה אינה ריקה.
4. היות וכל איבר בתמונה הוא מסוג uint8 שיכול לייצג מספרים בטווח 0-255 בלבד, והיות ומיקום התחלת ההודעה נשתל בעצמו כפיקסל של התמונה, ניתן לשתול את ההודעה רק באינדקסים הקטנים מ-256.
5. ניתן להניח כי מיקום ההודעה לא יהיה כזה שבו נתוני ההודעה ידרסו את אחד משלושת הפיקסלים האחרונים בשורה האחרונה, מה שהיה עלול לפגוע במידע הנוסף.

1 ממשו פונקציה **arr_dist(a1, a2)** המקבלת שני מערכים ומחשבת מרחק בין שני המערכים כסכום של ערכים מוחלטים של הפרשים בין האיברים מתאימים. יש להניח שגודל המערכים זהים. יש לכתוב את הפונקציה ללא שימוש בלולאות, בשורה אחת בלבד.

דוגמה: המרחק בין המחרוזת "Hello" והמחרוזת "Halmo" הוא 5 .ערכי הASCII של התוים מופיעים בסוגריים.

H (72)	**e** (101)	**l** (108)	**l** (108)	**o** (111)
H (72)	**a** (97)	**l** (108)	**m** (109)	**o** (111)
72-72	|101-97|	108-108	|108-109|	111-111
0	4	0	1	0

```
>>> a1 = ascii_to_np_array('Hello')

>>> print(a1)

[72 101 108 108 111]

>>> a2 = ascii_to_np_array('Halmo')
```

```
>>> print(arr_dist(a1, a2))
```

5

:2

ממשו פונקציה **find_best_place(im, np_msg)** המקבלת תמונה ומערך ההודעה ומחזירה זוג אינדקסים (שורה ועמודה, כtuple) של המיקום בתמונה שבו מתחיל רצף ערכים (באותו אורך כמו של אורך ההודעה) שהכי דומה למערך ההודעה. כלומר, ככל שהמרחק בין רצף הערכים לבין ההודעה (כפי שמתקבל מהפונקציה בשאלה 1) יותר קטן, הוא יותר דומה לה.

מותר להשתמש בלולאות.

זכרו כי יש לוודא שההודעה מוכלת כולה בשורה אחת של התמונה (ראו תאור האלגוריתם למעלה).

דוגמה:

```
>>> im1 = imageio.imread('parrot.png')
>>> np_msg = ascii_to_np_array('Hello')
>>> idx = find_best_place(im1, np_msg)
>>> print(idx, type(idx))
```

(110, 121) <class 'tuple'>

:3

ממשו פונקציה **create_image_with_msg(im, im_idx, np_msg)** המקבלת תמונה, זוג האינדקסים של מיקום בתמונה ואת מערך ההודעה. הפונקציה תחזיר מערך דו מימדי המייצג את תמונת הפלט בו שתולה ההודעה.

הפונקציה תבצע את השלבים הבאים:

1 .ייצור עותק של מערך התמונה (ניתן להשתמש במתודה copy הפועלת על מערך של numpy) על מנת שמערך הקלט המקורי לא ישתנה.

2 .העתקת מערך ההודעה למיקום המוגדר ע"י האינדקסים idx_img (שורה ועמודה, כtuple).

3 .שמירת ערכו של אינדקס השורה בה מתחילה ההודעה במיקום [-1, -3] בתמונה החדשה.

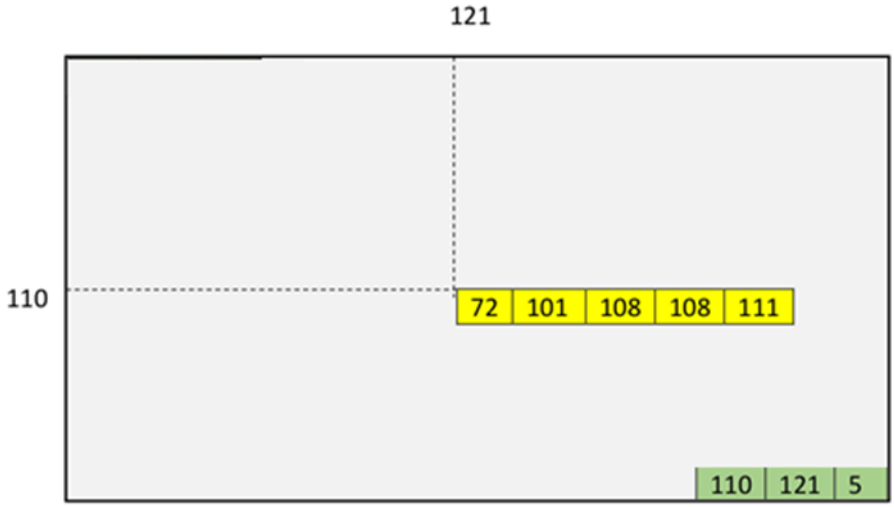
4 .שמירת ערכו של אינדקס העמודה בה מתחילה ההודעה במיקום [-1, -2] בתמונה החדשה.

5 .שמירת אורך ההודעה במיקום [-1,-1] בתמונה החדשה.

6 .החזרת המערך הדו מימדי שמייצג את התמונה חדשה.

אין להשתמש בלולאות.

דוגמה: בהמשך לדוגמה של שאלה 2 ,הפונקציה תשתול את ההודעה באופן הבא בתמונת הקלט:



הפיקסלים הצהובים (בתוך התמונה) – תוכן ההודעה, פיקסלים הירוקים בפינה הימנית התחתונה – – מידע עזר, ושאר הפיקסלים ישארו כמו בתמונה המקורית.

:4

ממשו פונקציה **put_message(im, msg)** המקבלת תמונה ומחרוזת ההודעה ומחזירה תמונה עם מחרוזת נסתרת בפנים. אין להשתמש בלולאות. הפונקציה מבצעת את השלבים הבאים:

1. הופכת את המחרוזת למערך.
2. מוצאת את המקום הכי מתאים בתמונה (באמצעות הפונקציה של שאלה 2).
3. בונה תמונה חדשה (באמצעות הפונקציה של שאלה 3).
4. מחזירה תמונה חדשה.

:5

ממשו פונקציה **get_message(im)** שמחזירה את טקסט ההודעה השתולה בתמונת הקלט. יש להניח שהודעה קיימת. אין להשתמש בלולאות.

הפונקציה מבצעת השלבים הבאים:

1. שולפת פיקסלים של הודעה כמערך באמצעות שלושת הפיקסלים השתולים כעזר בתמונה.
2. מחזירה את מחרוזת הטקסט של ההודעה השתולה (יש לקרוא לפונקצית **np_array_to_ascii** הממומשת).

שאלה 2 - pandas

"גראלט מריוויה ("Geralt of Rivia") מסתובב בכל רחבי היבשת במצוד אחר מפלצות המטרידות את תושבי הממלכות. כוויצ'ר("witcher") יפה תואר אך חסר רגשות, גראלט בוחר את יעדו לפי חישוב קר של עלות מול תועלת.

https://en.wikipedia.org/wiki/The_Witcher_(TV_series)

במהלך השאלה, נעזר בקובץ המשימות המכיל את כלל היעדים והמפלצות אותם יש לחסל. הקובץ בפורמט CSV ומכיל את המידע הבא:

- בשורה הראשונה מופיעים שמות עמודות המידע בסדר הבא:
  - Kingdom - העמודה הראשונה, מכילה את שם הממלכה במצוקה.
  - Bounty - העמודה השנייה, מכילה את הגמול על חיסול המפלצת.
  - Expenses - העמודה השלישית, מכילה את עלות המסע של גראלט לאותה ממלכה.
  - Duration - העמודה הרביעית מכילה את מספר הימים שייקח לגראלט להשלים את המשימה.
- שאר השורות מכילות את המידע הרלוונטי בהתאם לעמודות כמצוין לעיל (ראו את טבלת הדוגמא למטה)
- לנוחיותכם הטבלה הבאה נמצאת כקובץ בשם "missions.csv" בפורמט csv בין קבצי התרגיל שקיבלתם:

Kingdom	**Bounty**	**Expenses**	**Duration**
Temeria	1000	250	5
Redania	1500	500	3
Kaedwen	500	100	7
Cintra	2500	2000	3

א. ממשו את הפונקציה:
read_missions_file(file_name)

- הפונקציה מקבלת את שמו של קובץ המידע כפי שהוגדר לעיל (מחרוזת)
- הפונקציה תבנה dataframe של החבילה **pandas** כך ש:
  - הטבלה בעלת 3 עמודות: Bounty, Expenses, Duration.
  - שורות הטבלה בעלות שמות הממלכות, כלומר הindex של הטבלה הוא לפי Kingdom.
- במקרה של **שגיאת IO**, יש לתפוס את השגיאה ולהעלות שגיאת IO עם הכיתוב "An IO error occurred" (ראו דוגמת הרצה בעמוד הבא).
- ניתן להניח כי אם ניתן שם קובץ קיים, ערכיו יהיו תקינים ומכילים לפחות שורת משימות אחת (ושורת כותרות לעמודות).
- רמז: ניתן (אך לא חייבים) להיעזר בפקודה pd.read_csv על מנת לטעון את הקובץ לטבלה של pandas. הסתכלו בתיעוד הפונקציה באינטרנט בכדי להעביר את הארגומנטים המתאימים להשלמת השאלה.

ב. ממשו את הפונקציה:

add_daily_gain_col(bounties)

- קלט הפונקציה הינו טבלת pandas המייצג את טבלת המשימות כפי שיוחזר בסעיף א'.
- על הפונקציה להוסיף עמודה המכילה את הרווח היומי הנקי בכל ממלכה. רווח יומי נקי נמדד לפי תגמול בניקוי ההוצאות חלקי מספר הימים שלוקח לבצעה.
- **לדוגמא:** בהינתן מערך כפי שמופיעה בטבלה לעיל, כדאיות המשימה בממלכת Temeria הינה 150 מכיוון ש-$\frac{1000-250}{5} = 150$
- העמודה החדשה תיקרא "Daily gain"
- **הפונקציה לא צריכה להחזיר כלום**

בכל הסעיפים הבאים הקלט לפונקציה הינו טבלת pandas המייצג את טבלת המשימות כפי שתוחזר בסעיף א'.

ג. ממשו את הפונקציה:

sum_rewards(bounties)

- על הפונקציה לחשב ולהחזיר את סכום הכסף שגראלט יקבל אם יבצע את **כלל** המשימות **בניקוי** הוצאות המסע.
- ניתן להניח כי הגמול ממשימה גדול מהוצאות המסע למשימה.
- לדוגמא, אם יבצע משימה במלכת Kaedwen, גראלט ירוויח 500, אך יאבד 100 על הוצאות מסע. בסה"כ ירוויח 400.
- **אין להשתמש בלולאות ויש לכתוב את גוף הפונקציה בשורה אחת.**

ד. ממשו את הפונקציה:

find_best_kingdom(bounties)

- על הפונקציה למצוא ולהחזיר את **שם הממלכה** הכדאית ביותר לגראלט. כדאיות משימה נמדדת לפי תגמול בניקוי ההוצאות חלקי מספר הימים שלוקח לבצעה (כמו שתואר בסעיף ב')
- ניתן להניח כי כל ממלכה מופיעה פעם אחת בלבד בקובץ וכי יש ממלכה הכי כדאית אחת ויחידה.
- **לדוגמא:** בהינתן מערך כפי שמופיעה בטבלה לעיל, כדאיות המשימה בממלכת Temeria הינה 150 מכיוון ש-$\frac{1000-250}{5} = 150$
- ניתן להשתמש בסעיף ב' על מנת לחשב את העמודה-"Daily gain" ולהשתמש בה כדי למצוא את הממלכה הכי כדאית.
- **אין להשתמש בלולאות ורצוי, אך לא חובה, לבצע את החישוב בשורה אחת (לאחר השימוש בסעיף ב').**
- **שימו לב - אין להניח שסעיף ב' בוצע לפני הכניסה לפונקציה. במידה ומשתמשים בעמודה שהוא מייצר יש לוודא שקיימת (ניתן פשוט לקרוא לו בתחילת הפונקציה)**

דוגמת הרצה:

סעיף א'- קלט לא תקין (קובץ שלא קיים)

```
>>> read_missions_file("notAFile.csv")
Traceback (most recent call last):
  File "C:\Users\LENOVO\Desktop\python2021\ex10\EX_10_sol.py", line 67, in read_missions_file
    [redacted]
  File "C:\Users\LENOVO\AppData\Local\Programs\Python\Python37-32\lib\site-packages\pandas\io
\parsers.py", line 686, in read_csv
    return _read(filepath_or_buffer, kwds)
  File "C:\Users\LENOVO\AppData\Local\Programs\Python\Python37-32\lib\site-packages\pandas\io
\parsers.py", line 452, in _read
    parser = TextFileReader(fp_or_buf, **kwds)
  File "C:\Users\LENOVO\AppData\Local\Programs\Python\Python37-32\lib\site-packages\pandas\io
\parsers.py", line 936, in __init__
    self._make_engine(self.engine)
  File "C:\Users\LENOVO\AppData\Local\Programs\Python\Python37-32\lib\site-packages\pandas\io
\parsers.py", line 1168, in _make_engine
    self._engine = CParserWrapper(self.f, **self.options)
  File "C:\Users\LENOVO\AppData\Local\Programs\Python\Python37-32\lib\site-packages\pandas\io
\parsers.py", line 1998, in __init__
    self._reader = parsers.TextReader(src, **kwds)
  File "pandas\_libs\parsers.pyx", line 382, in pandas._libs.parsers.TextReader.__cinit__
  File "pandas\_libs\parsers.pyx", line 674, in pandas._libs.parsers.TextReader._setup_parser
_source
FileNotFoundError: [Errno 2] No such file or directory: 'notAFile.csv'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    read_missions_file("notAFile.csv")
  File "C:\Users\LENOVO\Desktop\python2021\ex10\EX_10_sol.py", line 69, in read_missions_file
    raise IOError("An IO error occurred")
OSError: An IO error occurred
```

סעיף א' – קובץ דוגמה (מצורף לתרגיל)

```
>>> filename = "missions.csv"
>>> df = read_missions_file(filename)
>>> print(df)
         Bounty  Expenses  Duration
Kingdom
Temeria    1000       250         5
Redania    1500       500         3
Kaedwen     500       100         7
Cintra     2500      2000         3
```

שאר הסעיפים מבוססים על הטבלה הנ"ל

סעיף ב'

```
>>> add_daily_gain_col(df)
>>> print(df)
         Bounty  Expenses  Duration  daily gain
Kingdom
Temeria    1000       250         5  150.000000
Redania    1500       500         3  333.333333
Kaedwen     500       100         7   57.142857
Cintra     2500      2000         3  166.666667
```

סעיף ג'

```
>>> sum_rewards(df)
2650
```

סעיף ד'

```
>>> print(find_best_kingdom(df))
Redania
```