מבוא למדעי המחשב 67101

תרגיל 10 - רשת ויקיפדיה

להגשה בתאריך 18/01/2016 בשעה 22:00

בתרגיל זה נבנה רשת שתייצג את מערכת המאמרים בויקיפדיה. כל מאמר ייצג קודקוד ברשת וקבוצת כל הלינקים בין המאמרים השונים תהווה את צלעות הרשת. בתרגיל זה עליכם לממש את הפונקציות בשני קבצים: article.py ו- wikinetwork.py.

טרמינולגיה: לשם פשטות בתרגיל זה נקרא למאמרים שמאמר A מפנה אליהם "שכנים שיוצאים מA" ולמאמרים שמפנים לA נקרא "שכנים שנכנסים לA".

משימה 1 - בניית רשת ויקיפדיה

במשימה זאת עליכם לבנות את מבנה הנתונים של רשת ויקיפדיה.

ראשית תסתייעו בקובץ אשר מכיל חלק ממבנה הרשת ויקיפדיה באנגלית. המידע מוצג ע"י רשימה של זוגות מאמרים על פי הפורמט הבא:

articleA	articleB
articleA	articleC
articleC	articleH
⋮	

כאשר כל זוג מאמרים מופרד **בטאב** (לא מופרד ברווח).

קובץ כזה לדוגמא נתון לכם באתר בשם links.txt.

כאשר הרשומה articleA articleB מציינת שיש הפניה (לינק) מהמאמר שכותרתו היא articleA למאמר שכותרתו היא articleB (שימו לב שזה לא מחייב שתהיה הפניה מ articleB ל articleA).

בקובץ **wikinetwork.py** כתבו פונקציה:

```
def read_article_links(file_name):
```

אשר מקבלת את שם הקובץ ומחזירה רשימה שאיבריה הן כל זוגות המאמרים, כאשר כל זוג הוא רשומה (tuple) של שני שמות המאמרים (כלומר בכל רשומה שני איברים).

כעת נבנה שתי מחלקות, המחלקה הראשונה תייצג מאמרים ברשת (Article), והמחלקה השנייה תייצג את הרשת עצמה (WikiNetwork) ותכיל בתוכה איברים מסוג Article. אנחנו מזכירים לכם כי למדתם בכיתה אודות אנקפסולציה של אובייקטים ושימוש נכון בAPI של מחלקות. בתיאור התרגיל אנחנו נגדיר את המתודות שעליכם יהיה לבנות עבור שתי המחלקות - מתודות אלו יגדירו את הAPI של המחלקות. מותר לכם ליצור מתודות נוספות אך אסור למחלקות האחרות לעשות בהן שימוש, על כן עליהן להיות מוגדרות באופן פרטי.

תחילה נבנה את המחלקה שתייצג אובייקטים מסוג **Article**.

המחלקה תבנה כולה בקובץ **article.py**.

טיפ: קראו קודם את כל המתודות של האובייקט לפני שאתם ניגשים ליישם את הבנאי.

הבנאי של המחלקה יקבל כארגומנט את כותרת המאמר (כמחרוזת) ויצור ממנו אובייקט מסוג Article. על אובייקטים אלו לשמור collection (רשימה\רשומה\קבוצה\מילון) שיכיל את המאמרים השכנים היוצאים של המאמר המיוצג באובייקט. הcollection יכיל אובייקטים נוספים מסוג Article.

```
class Article:
      def __init__(self, article_title):
```

בנוסף ממשו את המתודות הבאות של המחלקה Article:

```
     def get_title(self):
```

אשר מחזירה את כותרת המאמר.

```
     def add_neighbor(self, neighbor):
```

אשר מוסיפה אובייקט `Article` לcollection של המאמר.

```
     def get_neighbors(self):
```

אשר מחזירה את שכניו של המאמר בפורמט של רשימה (רשימת אובייקטים מסוג Article).
שימו לב כי זו מתודה מסוג "getter", ולכן עליכם לוודא שאין ביכולתו של המשתמש באובייקט לשנות את שכניו של המאמר לאחר שימוש בה.
וכן,

```
     def __repr__(self):
```

אשר מחזירה מחרוזת שמייצגת את המאמר. נבנה מחרוזת זו באופן הבא: ראשית המחרוזת תכיל את כותרת המאמר, לאחריו פסיק, ולאחריו רשימת שמות השכנים היוצאים של המאמר. נרצה שהמחרוזת תדמה לפורמט של tuple ועל כן המחרוזת כולה תוקף בסוגריים עגולות. לדוגמא, אם 'a' הוא שם מאמר ולו שכנים יוצאים 'b' ו'c' אז נסמן את האיבר ב

```
('a', ['b', 'c'])
```

וכן את המתודות:

```
     def __len__(self):
```

אשר מחזירה את מספר השכנים היוצאים של המאמר.

```
     def __contains__(self, article):
```

אשר מקבלת **אובייקט** של מאמר ומחזירה האם **המאמר** הוא שכן יוצא של self.
כאמור, אתם רשאים להגדיר מתודות **פרטיות** נוספות בקובץ זה.

כעת נגיע לחלק העיקרי של בניית הרשת.
לצורך זה עליכם לייצר מחלקה בשם `WikiNetwork` שתכיל את המאמרים ברשת. המאמרים ישמרו ברשת בתור אובייקטים של המחלקה Article. המחלקה WikiNetwork וכל המתודות שלה יוגדרו בקובץ **wikinetwork.py**.

גם במחלקה זו, תחילה קראו את כל משימותיה ורק לאחר מכן התחילו בבניית הבנאי.
המחלקה `WikiNetwork` נראית כך:

```
class WikiNetwork:
```

בניית הרשת תעשה ע"י הבנאי:

```
   def __init__(self, link_list):
```

אשר מקבל רשימת **כותרות** של זוגות מאמרים בדומה לפורמט שנוצר ע"י הפונקציה `read_article_links`, ומייצר אובייקט מסוג WikiNetwork. על כל אובייקט כזה להחזיק collection (כלומר רשימה/מילון/קבוצה) של מאמרים (מסוג Article שהוגדר למעלה). הcollection צריך להכיל את כל המאמרים בקובץ (בין אם הם מופיעים כשכנים נכנסים או יוצאים), בתור אוביקט מסוג Article, בדיוק פעם אחת. עליכם לוודא כי בסיום הבנאי, כל אובייקט מסוג Article ברשת מכיל את כל שכניו היוצאים (על פי רשימת הקלט) בcollection הפנימי שלו.
הנחיה: גישה לאיברים במילון (dictionary) בפייתון היא מהירה (נעשית בזמן קבוע, ללא תלות בגודל המילון) ולכן מומלץ להשתמש במילון בכדי לייעל את בניית הרשת.
חשבו כיצד עליכם ליצר את הרשת באופן יעיל ולוודא כי הרשת מכילה בסופו של הבנאי את כל המאמרים ולכל אחד מהם גם את כל שכניו.

כעת ממשו את המתודות הבאות של `WikiNetwork`:

```
def update_network(self, link_list):
```

המקבלת רשימה של זוגות של מאמרים (בדומה למה שמחזירה הפונקציה `read_article_links`) ומעדכנת לפיה את הרשת, בדומה למה שהוגדר ב__init__. שימו לב, פונקציה זאת לא מהווה תחליף ל__init__, באופן זה שהיא לא מייצרת רשת חדשה, אלא רק מעדכנת את האובייקט עליו נקראה הפונקציה.

כמו כן ממשו את שאר המתודות של של `WikiNetwork`:

```
def get_articles(self):
```

אשר מחזירה רשימה של כל ה- Articles ברשת (כלומר רשימה של אובייקטים מסוג Article). הזכרו בהערה לעיל בנוגע למטודות מסוג "getter".
וכן את,

```
def get_titles(self):
```

שמחזירה את רשימת כותרות כל המאמרים ברשת (רשימה של מחרוזות).

```
def __contains__(self, title):
```

שמחזירה האם מאמר נמצא ברשת מאמר **לפי כותרתו**.

```
def __len__(self):
```

שמחזירה את מספר המאמרים ברשת.

```
def __repr__(self):
```

אשר מחזירה מחרוזת שמייצגת את הרשת. הפלט צריך להיות מחרוזת המייצגת מילון שמפתחותיו הם שמות של מאמרים, וערכיו הם הייצוג הטקסטואלי של מאמרים (כמו שהוגדר לעיל במחלקה Article). לדוגמא, רשת המכילה מאמרים בשמות 'a', 'b', 'c', יכולה להיות מיוצגת ע"י המחרוזת:

```
{'a': ('a', ['b', 'c']), 'b': ('b', ['a']), 'c': ('c', [])}
```

וכן,

```
def __getitem__(self, title):
```

אשר מקבלת כותרת של מאמר ומחזירה את אובייקט המאמר המתאים לשמו. הדבר הנכון לעשות במידה ולא נמצא מאמר עם שם זה הוא להעלות שגיאה ע"י כתיבת השורה הבאה:

```
raise KeyError(title)
```

התנהגות זו תואמת להתנהגות במילונים של פייתון, כשמבקשים ערך לפי מפתח שלא קיים במילון. כיון שלא לימדנו בהרצאות על העלאת שגיאה, אנחנו לא דורשים טיפול כלשהו במקרה שבו לא נמצא ברשת מאמר

מתאים. מומלץ לטפל במקרה זה כמו שצריך, אך אין חובה. אין צורך, וגם לא מומלץ, לטפל במקרה זה ע"י הדפסת ערך שגיאה, או החזרה מכוונת של ערך כלשהו.
ניתן להוסיף שדות פרטיים (משתנים) או מתודות פרטיות נוספות למחלקות Article ו-WikiNetwork.

משימות 2-5 מגדירות מתודות נוספות של המחלקה WikiNetwork.

<u>משימה 2 - Page Rank</u>
כאשר מבצעים חיפוש בגוגל, מתקבלת רשימת האתרים שמוצגת על פי סדר מסוים. קביעת הסדר נעשית בין היתר ע"י אלגוריתם Page Rank (שנקראת על שם Larry Page - אחד ממייסדיה של גוגל) אשר מאפשר לדרג "חשיבות" של האתרים ברשת בדרך מסוימת. הנחת האלגוריתם היא שלאתרים חשובים יש מספר רב יותר של לינקים שמפנים אליהם (לינקים נכנסים) מאשר לאתרים פחות חשובים. רעיון האלגוריתם הוא ניסיון להתחקות אחר משתמש אקראי ש"גולש" מאתר לאתר ע"י לחיצה על הלינקים.

האלגוריתם ניתן לתאור באופן הבא: לכל מאמר i בזמן t יש כמות כסף M^t, אשר מסמלת את מידת החשיבות שלו. בתחילה, לכל מאמר יש שקל אחד, כלומר $M^0(i) = 1$ לכל מאמר i. האלגוריתם הוא איטרטיבי וכל איטרציה מורכבת משני שלבים:

1. כל מאמר מחלק d (אשר מוגדר להיות d=0.9 כברירת מחדל) מהכסף שברשותו באופן שווה בשווה לכל המאמרים שהוא מפנה אליהם (כלומר $d\frac{M^t(A)}{OUT(A)}$ כאשר $M^t(A)$ הוא הכסף של מאמר A בזמן t, ו $OUT(A)$ הוא מספר השכנים היוצאים של A) ומהחלק הנותר הוא מחלק מהכסף לכל המאמרים ברשת (כלומר לא רק השכנים) באופן שווה.
2. כל מאמר אוסף את כל הכסף שקבל הן בשל היותו שכן יוצא של מאמרים והן בשל היותו מאמר ברשת. כל מאמר מעדכן את סכום הכסף שיהיה לו באיטרציה הבאה ע"י סיכוסשל סכום הכסף הנ"ל. שימו לב כי יישום נכון של האלגוריתם יבטיח כי בכל איטרציה סכום הכסף שמקבל מאמר A רק בשל היותו מאמר ברשת הוא בדיוק $1-d$ (ראו משוואות למטה) על כן: **אין צורך** לחשב את החלק שנותר מכל המאמרים. ניתן למעשה להניח שסכום הכסף שיש לכל מאמר בסוף כל איטרציה הוא $1-d$ פלוס הכסף שקבל משכניו.

ניתן לתאר את סכום הכסף שמקבל מאמר A בזמן t+1 כפונקציה של סכום הכסף של המאמרים בזמן t באופן הבא:

$$M^{t+1}(A) = d(\frac{M^t(B)}{OUT(B)} + \frac{M^t(C)}{OUT(C)} + \ldots) + \frac{1-d}{n}(M^t(A) + M^t(B)\ldots)$$

כאשר הסוגריים הראשונים מייצגים את הכסף שקבל המאמר A משכניו הנכנסים והסוגריים השניים את הכסף שקבל רק בשל היותו מאמר ברשת.

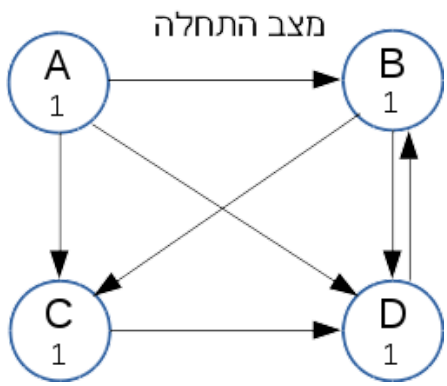
$$= d\sum_{i\in N(A)} \frac{M^t(i)}{OUT(i)} + \frac{1-d}{n}\sum_i M^t(i)$$

$$= d\sum_{i\in N(A)} \frac{M^t(i)}{OUT(i)} + (1-d)$$

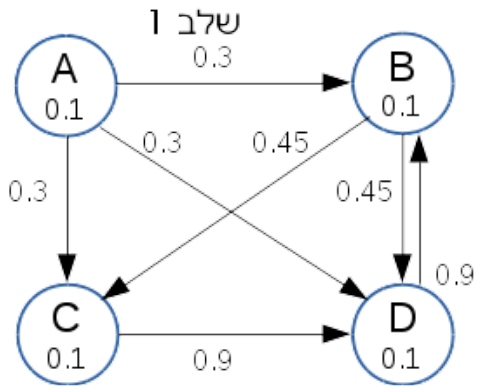
כאשר N(A) היא קבוצת השכנים הנכנסים של A, ו OUT(i) מסמן את מספר השכנים היוצאים של מאמר i, ו $M^t(i)$ מסמן את כמות הכסף של מאמר i בזמן t, ו-n זה מספר המאמרים ברשת. המעבר מהמשוואה

השנייה לשלישית הוא נכון כיוון שסכום הכספים של כל המאמרים בכל איטרציה הוא n. מכיוון ש $1-d=0.1$ הוא סכום קבוע נקרא לו "היתר".

נציג דוגמא: להלן רשת של ארבעה מאמרים. כל עיגול מייצג מאמר וחץ מעיגול i לעיגול j מייצג הפניה ממאמר i למאמר j.



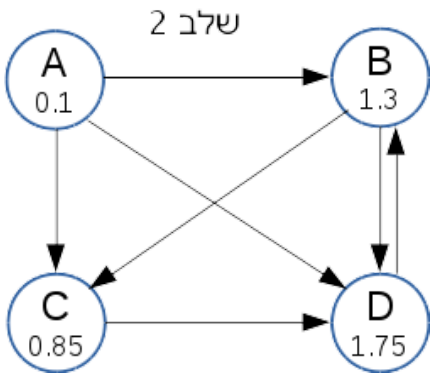
מצב התחלה - כל המאמרים מתחילים עם שקל בודד.

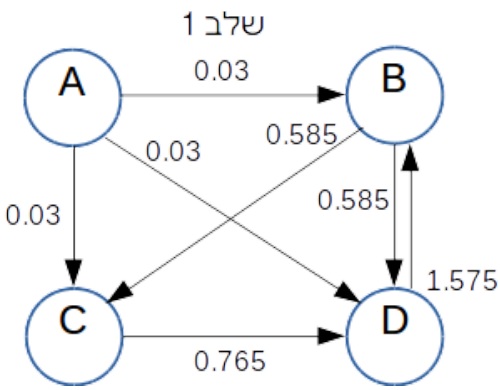


איטרציה 1, שלב 1 - כל מאמר מחלק 0.9 שקלים (כיוון שבחרנו d=0.9) באופן שווה לכל שכניו. בהתאם לכך A מחלק 0.3 לכל אחד משלושת משכניו, מאמר B מחלק 0.45 לכל אחד משני שכניו, ומאמרים C ו D מעבירים 0.9 במלואם לשכן הבודד שלהם. באיור מסומן 0.1 בכל מאמר (היתר) שכן כל מאמר מחלק $1-d=0.1$ לשאר המאמרים (לא רק שכנים),וכפועל יוצא מכך הוא שבשלב 2 מתווסף לכל מאמר 0.1.

איטרציה 1, שלב 2 - הסכומים של המאמרים הבאים מצטברים באופן הבא:

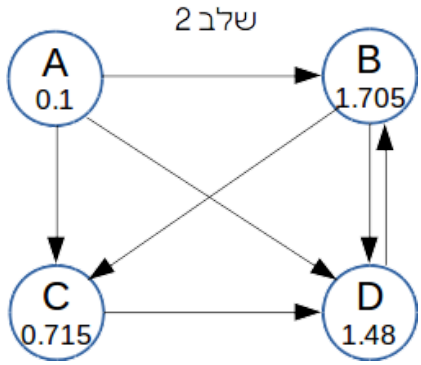
- מאמר A נותר עם יתר של 0.1 משום שאין לו שכנים נכנסים ועל כן לא קבל כסף מעבר ליתר.
- מאמר B מקבל 0.3 מA, ו0.9 מD. מכיוון שB מקבל בנוסף יתר 0.1, כעת יש לB בסה"כ 1.3 שקלים.
- מאמר C מקבל 0.3 מA, ו0.45 מB. מכיוון שC מקבל בנוסף יתר 0.1, כעת יש לC בסה"כ 0.85 שקלים.
- מאמר D מקבל 0.3, מקבל 0.45 מB ו0.9 מC. מכיוון שD מקבל יתר 0.1, כעת יש לD בסה"כ 1.75 שקלים.





איטרציה 2, שלב 1 - כל מאמר i מחלק באופן שווה לכל שכניו (כלומר מחלק את הכסף שצבר באיטרציה 1 כפול 0.9). בהתאם לכך A מחלק 0.03 לכל אחד משלושת משכניו, מאמר B מחלק 0.585

לכל אחד משני שכניו, מאמר C מעביר 0.765 לD, ו-D מעביר 1.575 לB. בנוסף כל מאמר מחלק שווה בשווה בין כל המאמרים 10% (0.1) מהכסף שברשותו (כאמור אין צורך לחשב שלב זה, כפי שראינו לעיל בסיום כל איטרציה היתר שמקבל כל מאמר הוא זהה).



איטרציה 2, שלב 2- הסכומים של המאמרים הבאים מצטברים באופן הבא:

- מאמר A נותר עם יתר של 0.1 משום שאין לו אף שכן נכנס שמעביר לו כסף
- מאמר B מקבל 0.03 מA, ו1.575 מD. מכיוון שB מקבל בנוסף יתר של 0.1, כעת יש לB בסה"כ 1.705 שקלים
- מאמר C מקבל 0.03 מA, ו0.585 מB. מכיוון שC מקבל בנוסף יתר 0.1, כעת יש לC בסה"כ 0.715 שקלים
- מאמר D מקבל 0.03, מקבל 0.585 מB ו0.765 מC. מכיוון שD מקבל בנוסף יתר 0.1, כעת יש לD בסה"כ 1.48 שקלים.

כמובן שבסופה של כל איטרציה סך סכומי הכסף ברשת זהה (בדיוק N).
הפונקציה תתכנס בסופו של דבר לפתרון הבא:
['D (=1.55379609544)', 'B (=1.5284164859)', 'C (=0.817787418655)', 'A (=0.1)']
במקרה הנ"ל, האתרים מתמיינים על פי מספר השכנים, אך לא תמיד זה המצב. נסו לחשוב האם ניתן ליצור סידורים אחרים, ומהו אינפוט מוצלח לאתר (כמה שיותר לינקים? לינקים רק מאתרים מסוימים?)

ממשו את הפונקציה page_rank השייכת למחלקה WikiNetwork:

```
def page_rank(self,iters,d=0.9):
```

אשר מקבלת מספר שלם iters שמייצג את מספר האיטרציות ש-pagerank מופעל, ומספר ממשי d בין 0 ל-1 , שמייצג את גודל התרומה למאמרים השכנים. הפונקציה מריצה את האלגוריתם ולאחר מכן מחזירה את **רשימת כותרות המאמרים** בסדר ממוין מהגדול לקטן, לאחר ביצוע page rank. במידה וישנו יותר ממאמר אחד עם אותו דירוג, אז החזירו אותם לפי סדר לקסיקוגרפי מהקטן לגדול. לדוגמא, במידה ותוצאות האלגוריתם היא הצמדים הבאים של שמות וציונים

```
[('aaa',0.8),('a',0.6),('aba',0.8),('b',2)]
```

על פלט הפונקציה להיות:

```
['b','aaa','aba','a']
```

כתבו את שלושת המאמרים בקובץ links עם הpage rank הגבוה ביותר בREADME עם הפרמטרים d=0.9 ו iters=50 (למרות שגם עבור מספר הרבה יותר קטן של איטרציות נקבל את אותה תוצאה).

משימה 3 - אינדקס ג'אקרד

אינדקס ג'אקרד הוא מדד לקרבה של שתי קבוצות אשר מוגדר ע"י גודל החיתוך של שתי קבוצות חלקי גודל האיחוד שלהן כלומר עבור קבוצות A וB אינדקס ג'אקרד שלהם מוגדר כך: $\frac{|A \cap B|}{|A \cup B|}$.

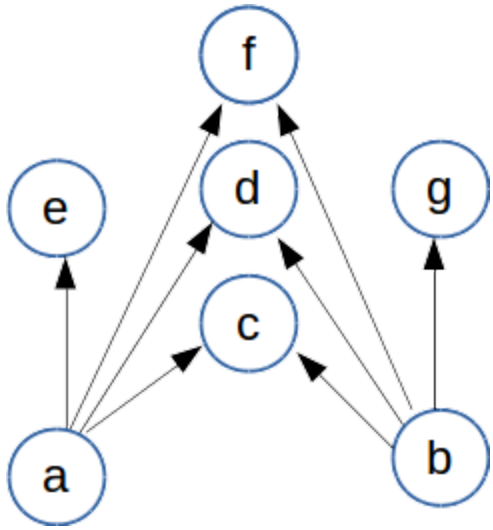
החיתוך של מאמרים שכותרתם a וb הוא קבוצת כל המאמרים שa **וגם** b מפנים אליהם. כלומר אם נגדיר את N(a) להיות השכנים היוצאים מ-a, אזי שהחיתוך של a וb הוא קבוצת המאמרים שנמצאים **גם** בN(a) **וגם** בN(b). האיחוד של שני מאמרים הוא קבוצת כל המאמרים שa **או** b **או שניהם** מפנים אליהם (כל המאמרים בN(a) ו/או ב-N(b)).

דוגמא (ראו איור): ממאמר a יוצאים שכנים c,d,e,f וממאמר B יוצאים שכנים c,d,f,g. במקרה זה החיתוך

בינהם הוא c,d,f, כלומר גודל החיתוך הוא 3, והאיחוד שלהם הוא c,d,e,f,g, כלומר גודל האיחוד הוא 5. לכן אינדקס ג'אקרד של מאמר a ומאמר b הוא 3/5.
קצת אינטואיציה: אינדקס ג'אקרד מסמן קירבה בין שני מאמרים כך שלמאמרים "דומים" יש יחסית הרבה שכנים יוצאים משותפים לשניהם (ולכן החיתוך), מצד שני ככל שלמאמר יש יותר שכנים יוצאים יש לו סיכוי רב יותר לשכנים משותפים עם מאמר אחר ללא קשר לדמיון ביניהם, ולכן מנרמלים (מחלקים) באיחוד השכנים היוצאים (ראו איור למטה).
ממש את הפונקציה jaccard_index השייכת למחלקה WikiNetwork:

```
def jaccard_index(self, article_title):
```

שמקבלת שם של מאמר, מחשבת את אינדקס הג׳אקרד של כל המאמרים ברשת ביחס למאמר , ומחזירה רשימה ממוינת של כל **כותרות המאמרים ברשת**, הממוינים לפי אינדקס הג'אקרד שחושב (כלומר האיבר הראשון ברשימה הוא המאמר בעל האינדקס ג'אקרד הגדול ביותר מבין כל המאמרים). שימו לב שאינדקס ג'אקרד של מאמר עם עצמו הוא 1. במקרה שבו יש כמה מאמרים בעלי אותו אינדקס ג'קארד, המאמרים הלולו יוחזרו ממוינים לפי סדר לקסיקוגרפי מקטן לגדול. במידה וכותרת המאמר לא נמצאת ברשת החזירו None. שימו לב כי אם למאמר אין צלע יוצאת אזי שאינדקס הג׳קארד שלו עם עצמו ידרוש חלוקה ב0. על כן במידה ולמאמר הקלט אין אף צלע יוצאת החזירו גם כן None.
מה האינדקס ג'אקרד השני הגבוה ביותר (כלומר הגבוה ביותר שהוא לא המאמר עצמו) למאמרים "Louis_XIV_of_France", "The_Godfather", "The_Lord_of_the_Rings" ו- "The_Simpsons" ?
כתבו את התוצאות שלכם בREADME.



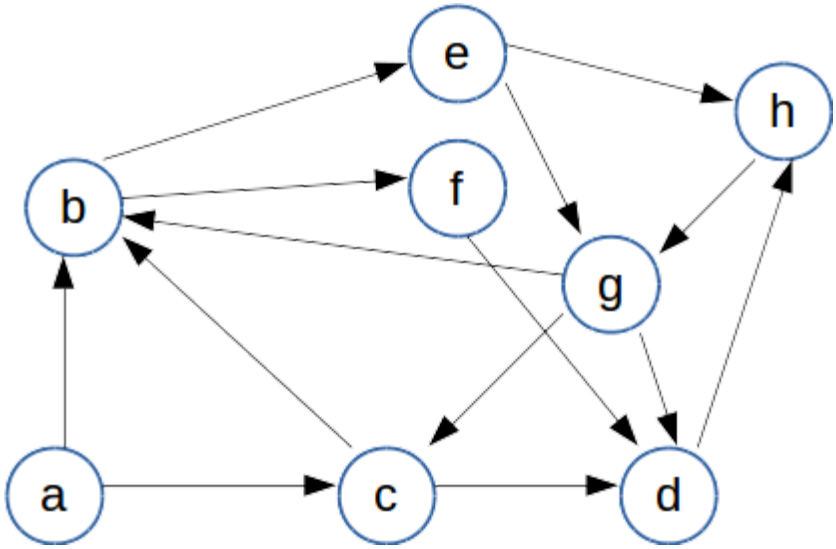
<u>משימה 4 - טיול ברשת</u>
נגדיר כי הרשת מכילה מעבר חוקי ממאמר א' למאמר ב' אם קיימת הפניה ממאמר א' למאמר ב'. טיול ברשת הוא סדרה של מעברים חוקיים על מאמרים. נגדיר דרגת כניסה של מאמר א׳ כמספר הצלעות הנכנסות אליו (מספר המאמרים שיש להם הפניה למאמר א׳).
טיול עולה ברשת הוא מעבר חוקי על המאמרים כאשר בכל צעד המעבר נקבע על פי הקריטריון הבא:

1. נבחר את המאמר שלו דרגת הכניסה הגבוהה ביותר מבין כל המאמרים אליהם ניתן לעשות צעד חוקי.
2. במידה וקיימים מספר מאמרים (להם ניתן לעשות צעד חוקי), עם מספר זהה ומקסימלי של דרגות כניסה, אז נבחר את המאמר שלכותרת שלו יש את הערך הקטן ביותר לפי הסדר הלקסיקוגרפי.

הטיול מסתיים (זה לא מובטח) כאשר מגיעים למאמר שאין לו שכנים יוצאים. שימו לב שבכדי למצוא את דרגת הכניסה של אובייקט מאמר ברשת, יש צורך לעבור על כל מאמרי הרשת.
דוגמא: נתונה לנו הרשת להלן (באיור). נניח שאנחנו מתחילים את הטיול במאמר a. לאחר מכן ניתן לעבור למאמרים b וc. המאמר הבא יהיה b מכיוון שיש לו דרגת כניסה 3 ואילו למאמר c יש דרגת כניסה 1. לאחר שהגענו לb ניתן לעבור למאמרים e ו f. לשני המאמרים הללו אותה דרגת כניסה, ולפי הסדר הלקסיקוגרפי e הוא נמוך מf, ולכן הבא יהיה מאמר e. המאמר הבא שייבחר יהיה g (ע"פ דרגת כניסה) ולאחריו b (גם כן ע"פ דרגת כניסה) .
ממשו את הפונקציה:

```
def travel_path_iterator( self , article_title):
```

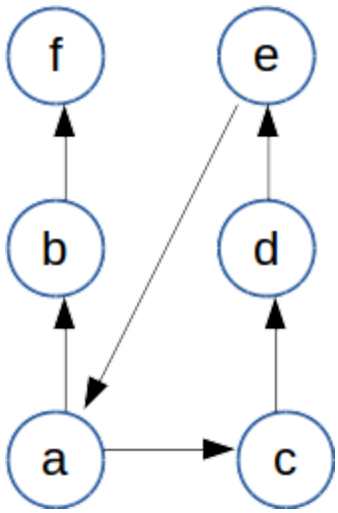
אשר מקבלת כותרת של מאמר ומחזירה גנרטור המחזיר לפי סדר את כותרות המאמרים בטיול. במידה והטיול מסתיים, הגנרטור יפסיק לייצר ערכים להחזרה. אין מניעה שהטיול ייכנס ללולאה אינסופית. במידה וכותרת המאמר לא קיים ברשת, יוחזר גנרטור של טיול עם מסלול ריק, שאינו מחזיר כותרות מאמרים. המאמר שאת כותרתו הפונקציה מקבלת צריך להיות המאמר הראשון במסלול.



<u>משימה 5 - חברים ממרחק d</u>
הגדרנו במשימה הקודמת כי מעבר ממאמר א' למאמר ב' הוא חוקי אם יש הפניה ממאמר א' למאמר ב'.
חבר ממרחק d של מאמר i, הוא מאמר אשר ניתן להגיע אליו ממאמר ב d מעברים חוקיים **לכל היותר**.
לדוגמא (ראו איור) קבוצת כל החברים ממרחק 2 של a, הם a,b,c,d,f. מאמר e אינו חבר ממרחק 2, משום שניתן להגיע אליו רק ע"י 3 מעברים חוקיים. נגדיר: חבר מדרגה 0 של מאמר הוא המאמר עצמו (ולכן מאמר הוא חבר של עצמו מכל דרגה).
ממשו את הפונקציה:

```
def friends_by_depth( self, article_title, depth):
```

אשר מקבלת כותרת של מאמר ומספר שלם חיובי ומחזירה את רשימת שמות כל החברים ממרחק d שלה.
במידה ושם המאמר לא נמצא ברשת הפונקציה מחזירה None.
רמז: מומלץ לבצע זאת ע"י שימוש בפונקציה פנימית אשר מקבלת קבוצת מאמרים A, ומחזירה את האיחוד של כל קבוצות השכנים של כל המאמרים A.



שאלות:
מה אחוז המאמרים מכלל המאמרים ברשת שהם במרחק 1 מ- United_States?
מה אחוז המאמרים מכלל המאמרים ברשת שהם במרחק 2 מ- United_States_dollar?
מה אחוז המאמרים מכלל המאמרים ברשת שהם במרחק 3 מ- Microsoft?
כתבו את התוצאה שקיבלתם בREADME

הוראות הגשה

בתרגיל זה עליכם להגיש את הקבצים הבאים:

1. article.py – עם המימוש שלכם למחלקה Article.
2. wikinetwork.py עם המימוש שלכם למחלקה wikinetwork ולפונקציה read_article_links.
3. README על פי הפורמט שמפורט בנהלי הקורס שכולל את התשובות לשאלות שבתרגיל.

יש להגיש קובץ zip הנקרא ex10.zip המכיל בדיוק את שלושת הקבצים הנ"ל.