# *שאילתות באינטרנט*

# *תרגיל 2 – דחיסת האינדקס*

## 1. תיאור התרגיל

בתרגיל הראשון נדרשתם לייצר מבני נתונים לצורך השאילתות השונות של ה IndexReader.

בתרגיל זה עליכם לממש דחיסה.

כדי לאפשר בדיקה אוטומטית התרגיל מגדיר בדיוק כיצד יהיה מבנה הקבצים.

הדחיסה עליה מדובר בתרגיל זה מתייחסת רק לשני האינדקסים המהופכים שבניתם האחד עבור מציאת reviews המכילים מילה מסוימת בשדה ה text והשני עבור מציאת ה reviews העוסקים במוצר (product) מסוים.

### 1.1. דחיסת המילון

דחיסת המילון רלוונטית אך ורק למילון של ה text שכן במילון של ה products כל ה terms במילון באותו אורך כך שאין משמעות לדחיסה.

עליכם לדחוס את קובצי המילון בשיטה של front coding 9 in 10.

כלומר, במחרוזת כל מילה עשירית תופיע בשלמותה ותשע המילים העוקבות לה יהיו מקודדות עם front-coding.

הטבלה תכיל שורה עבור כל בלוק. בכל שורה יהיה מצביע אל המקום במחרוזת בו נמצאת המילה הראשונה בבלוק (4 bytes). ובנוסף, עבור **כל מילה** בבלוק יופיעו הנתונים הבאים:

- אורך רשימת התפוצה (frequency) (4 bytes)
- מצביע לרשימת התפוצה (4 bytes)
- אורך המילה (למעט עבור המילה האחרונה בבלוק שאורכה יכול להיות מחושב בעזרת המצביע לבלוק הבא) ( 1 byte)
- גודל התחילית המשותפת עם המילה הקודמת (למעט עבור המילה הראשונה בבלוק שאורך התחילית שלה הוא 0) (1 byte)

סך הכל, גודל כל שורה בטבלה הוא: 102 bytes

המילון צריך להישאר בזיכרון בצורתו הדחוסה. כל חיפוש במילון יבצע חיפוש בינארי על הטבלה כדי למצוא את הבלוק בו נמצאת המילה ובתוך הבלוק יתבצע חיפוש סדרתי למצוא את המילה עצמה.

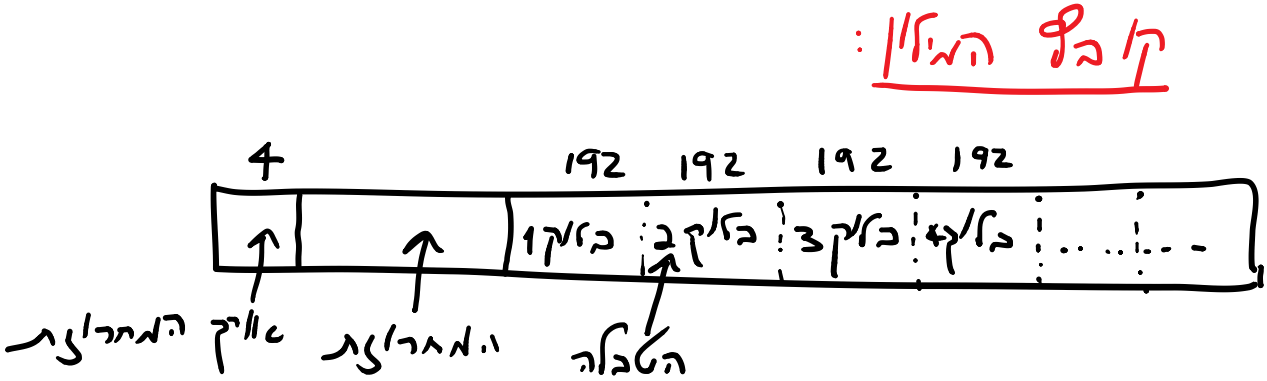
**<u>מבנה קובץ המילון:</u>**

**לקובץ המילון של ה text עליכם לקרוא בשם text.dic**

4 בתים ראשונים – אורך המחרוזת בבינארית.

המחרוזת : מקודדת ב ASCII

הטבלה : לכל בלוק 102 בתים המכילים את המידע כמתואר למעלה.




## 1.2. דחיסת רשימות התפוצה

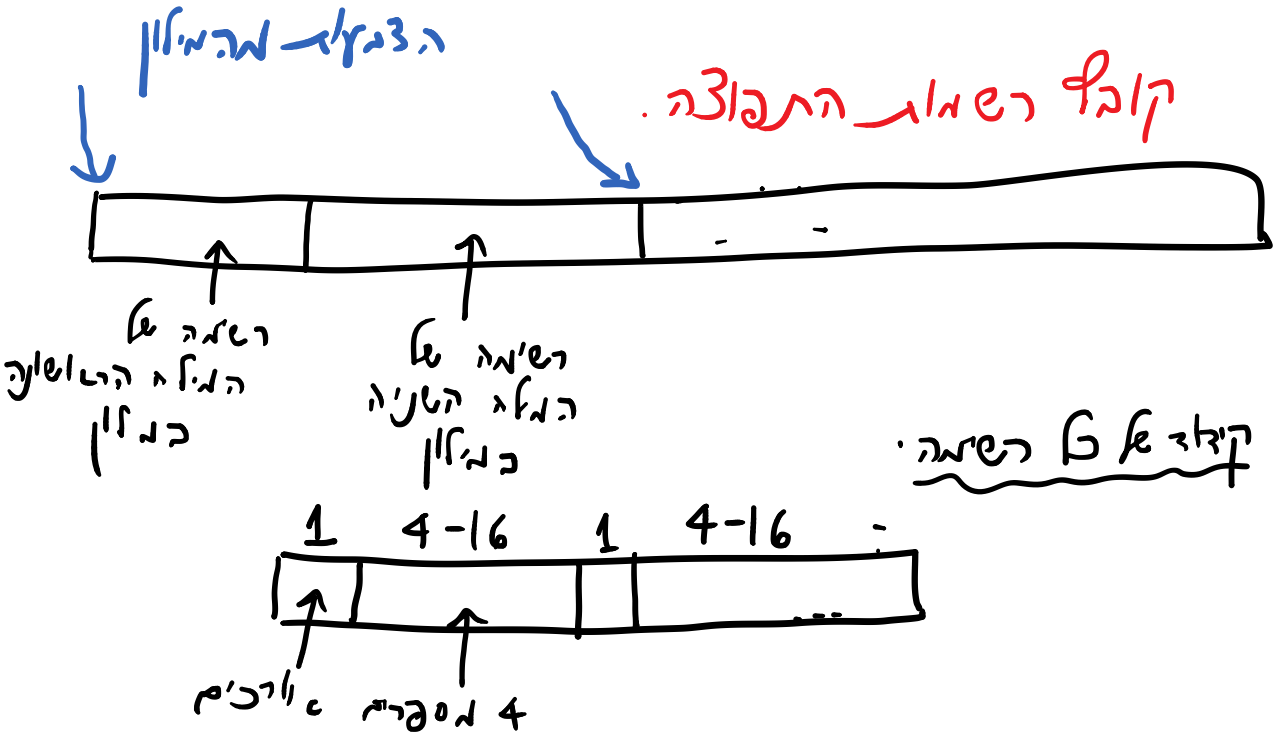
את רשימות התפוצה יש לדחוס בשיטת ה Group Varint בה לכל קבוצה של 4 מספרים יש byte נוסף המכיל את האורכים של המספרים.

זכרו לקודד את המרווחים בין מספרי ה reviews ולא את מספרי ה reviews עצמם וכן את ה frequency של המילה בכל מסמך.

בזמן שאילת שאילתה, עליכם למצוא במילון את המצביע לרשימת התפוצה הרלוונטית ולקרוא רק אותה מהדיסק אל הזיכרון. ה IndexReader יצטרך לפענח את הרשימה ולהחזיר את מספרי המסמכים בהם מופיעה המילה.

**לקובץ רשימות התפוצה של ה text עליכם לקרוא בשם text.pl**

**לקובץ רשימות התפוצה של ה products עליכם לקרוא בשם prod.pl**



## 2. דרישות הקוד

התכנית תכיל לפחות את שתי המחלקות הבאות: (ככל הנראה התכנית תכלול מחלקות נוספות הנחוצות לצורך מימושן)

2.1. CompressedIndexWriter: המחלקה תממש את אותו הממשק כמו הממשק של NonCompressedIndexWriter. ההבדל בין המחלקות הוא הצורה בה הקבצים נשמרים על הדיסק.

2.2. CompressedIndexReader: המחלקה תממש את אותו הממשק כמו הממשק של ה IndexReader מהתרגיל הקודם. ההבדל בין המחלקות הוא הצורך לפענח את הנתונים לפני מתן התשובה לשאילתות.

## 3. בדיקת התרגיל

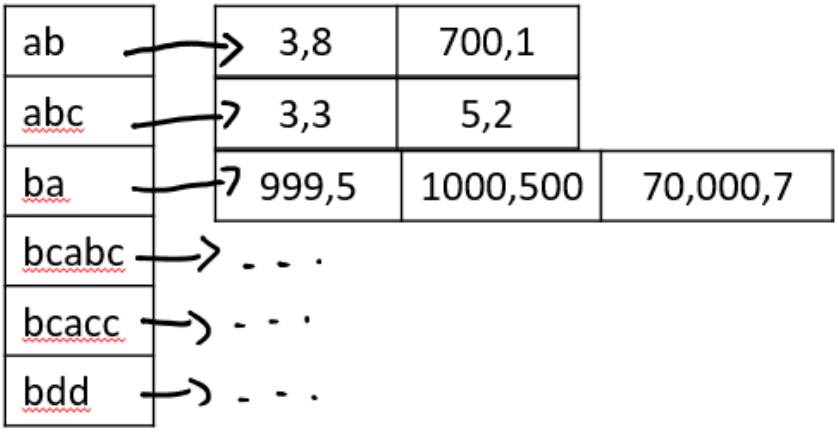
בבדיקת התרגיל ייעשה שימוש בערכת נתונים קטנה בסדרי גודל של הערכות הנמצאות באתר הקורס.

בדיקת התרגיל נעשית בעזרת מערכת אוטומטית. כדי שבדיקת התרגיל שלכם לא תיכשל (ותגרום להורדה בציון) הקפידו היטב על שמות המחלקות ועל מימוש ממשק זהה לזה שבתרגיל 1.

## 4. הגשת התרגיל

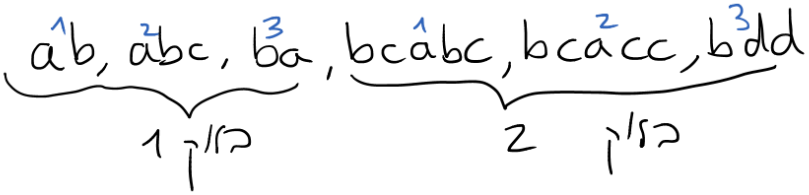
- התרגיל יוגש דרך אתר המכללה בפורמט ZIP.
- עבור כל זוג יש להגיש רק הגשה אחת. שם הקובץ צריך להיות ID1_ID2.zip כאשר ID1 ו ID2 הם מספרי הזהות של הסטודנטים המגישים את הפרויקט.
- **קבצי הקוד צריכים לכלול שני קבצים עם השמות CompressedIndexWriter.py ו CompressedIndexReader.py. תכנית הבדיקה תייבא (import) קבצים אלו כך שחשוב שתקפידו על השמות הנכונים (כולל אותיות גדולות וקטנות) קבצים אלו יכילו את המחלקות שנדרשתם לפתח.**
- **במידה והפרויקט שלכם מכיל קבצי קוד נוספים (מחלקות נוספות), באחריותכם לייבא אותם (import) מתוך הקבצים CompressedIndexWriter.py ו CompressedIndexReader.py.**
- יש לוודא כי הקובץ שהועלה הוא בפורמט הנכון ומכיל את כל הקבצים הרלוונטיים.

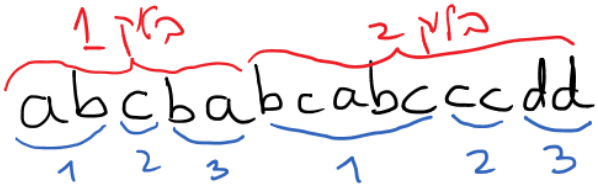
## דוגמה לקידוד




## קידוד המילון ב 2-in-3 front coding:

<u>קידוד המחרוזת:</u>



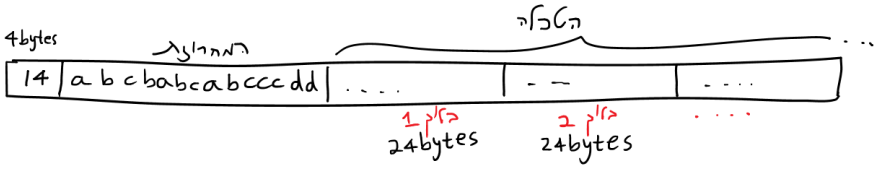




<u>קידוד הטבלה:</u>

| | Str-ptr | term1 | | | term2 | | | | term3 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | freq | post-ptr | length | freq | post-ptr | length | prefix | freq | post-ptr | prefix |
| block1 | 0 | 2 | 0 | 2 | 2 | 6 | 3 | 2 | 3 | 11 | 0 |
| block2 | 5 | ... | ... | 5 | ... | ... | 5 | 3 | ... | ... | 1 |

שדות ה- freq וה- post-ptr יכולים להתמלא רק אחרי
שמסיימים לבנות את ה posting-list של ה term.

## סך הכל הקובץ של המילון:




## קידוד רשימות התפוצה ב Group-Varint:

נקודד כל רשימה בפני עצמה. רשימה שמספר המספרים בה אינו מתחלק ב-4 נרפד בסוף בוספים כדי שכל רשימה תתחיל ברבי"ע חדשה.

| | | | |
|---|---|---|---|
| ab | 3,8 | 700,1 | |
| abc | 3,3 | 5,2 | |
| ba | 999,5 | 1000,500 | 70,000,7 |
| ... | | | |

רשימה של ab:

המספרים אותם צריך לקודד אחרי חישוב ה-Gaps:

| decimal | 3 | 8 | 697 | 1 |
|---|---|---|---|---|
| hex | 0x3 | 0x8 | 0x2B9 | 0x1 |
| length | 1 | 1 | 2 | 1 |

קידוד Group-Varint:

byte האורכים: 00000100 = 0x4

סה"כ הקידוד (בבתים):

0x04, 0x03, 0x08, 0x02, 0x B9, 0x1

אורך #1 #2 #3 #4

| | | | |
|---|---|---|---|
| ab | 3,8 | 700,1 | |
| abc | 3,3 | 5,2 | |
| ba | 999,5 | 1000,500 | 70,000,7 |
| ... | | | |

רשימה של abc:

המספרים אותם צריך לקודד אחרי חישוב ה-Gaps:

| decimal | 3 | 3 | 2 | 2 |
|---|---|---|---|---|
| hex | 0x3 | 0x3 | 0x2 | 0x2 |
| length | 1 | 1 | 1 | 1 |

קידוד Group-Varint:

byte האורכים: 00000000 = 0x0

סה"כ הקידוד (בבתים):

0x0, 0x3, 0x3, 0x2, 0x2

אורך #1 #2 #3 #4

| | | | |
|---|---|---|---|
| ab | 3,8 | 700,1 | |
| abc | 3,3 | 5,2 | |
| ba | 999,5 | 1000,500 | 70,000,7 |
| ... | | | |

רשימה של ba:

המספרים אותם צריך לקודד

אחרי חישוב ה-Gaps:

אוקי הרשימה אינו מתחלק ב-4 לכן מוסיפים 2 אפסים.

| decimal | 999 | 5 | 1 | 500 | 69000 | 7 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|
| hex | 0x3E7 | 0x5 | 0x1 | 0x1F4 | 0x10D88 | 0x7 | 0x0 | 0x0 |
| length | 2 | 1 | 1 | 2 | 3 | 1 | 1 | 1 |

קידוד Group-Varint:

byte האורכים הראשון: 01000001=0x41 , השני: 10000000=0x80

סה"כ הקידוד:

0x41, 0x3, 0xE7, 0x5, 0x1, 0x1, 0xF4, 0x80, 0x1, 0x0D, 0x88, 0x7, 0x0, 0x0

<u>סך הכל הקובץ של רשימות התפוצה:</u>

בדוגמה נראה רק את הרשימות של 3 המילים הראשונות. הקובץ ממשיך עם יתר רשימות התפוצה

0x04, 0x03, 0x08, 0x02, 0xB9, 0x1, 0x0, 0x3, 0x3, 0x2, 0x2

0x41, 0x3, 0xE7, 0x5, 0x1, 0x1, 0xF4, 0x80, 0x1, 0x0D, 0x88, 0x7, 0x0, 0x0

את המיקומים של הרשימות יש למלא בטבלה של המילון:

הרשימה של ab מתחילה במיקום 0

הרשימה של abc מתחילה במיקום 6

הרשימה של ba מתחילה במיקום 11