

מסנן בלום

תוכן עניינים

1	הקדמה	1
2	רקע תיאורטי	2
4	שימושים למסנן בלום	3
5	האלגוריתם עבור מסנן בלום	4
5	האלגוריתם לחישוב חוזק הסיסמא	5
6	ממשק התוכנה ורכיבו	6
9	ביבליוגרפיה	7

הקדמה:

הרעיון הומצא על ידי burton howard bloom בשנת 1970, בוגר מדעי המחשב בMIT אשר הגה את הרעיון בזמן עבודתו בחברת שימוש במחשבים.

מסנן בלום היא שיטה הנועדה לבדוק האם איבר נמצא בקבוצה קיימת של איברים, כאשר הוא לא שומר את האיברים עצם במבנה החיפוש אלא את ההאש שלהם. בניגוד לשיטות אחרות שיש יחס חח"ע בין תשובה שלילית \longleftrightarrow האיבר לא נמצא בקבוצה ותשובה חיובית \longleftrightarrow האיבר נמצא בקבוצה, במסנן בלום מתקיים עבור תשובה שלילית \longleftrightarrow האיבר לא נמצא בקבוצה ותשובה חיובית \longleftrightarrow אפשרי שהאיבר נמצא בקבוצה. כלומר החלפנו דיוק על חשבון שטח דיסק יותר קטן, למשל 15% מהשטח של השיטה חסינת השגיאות נדרש על מנת ליצור מסנן בלום ששולל 85% מהכניסות.

חשוב לציין שלמרות היתרון שנותן מסנן בלום יש לו מספר חסרונות, לא ניתן לגשת איבר עצמו בטבלה, אלא רק לבדוק אם או נמצא בטבלה, כמו כן מחיקה של איבר אינה אפשרית וכלל שמוסיפים איברים לטבלה אז הסיכוי לקבל תשובה חיובית למרות שהאיבר אינו נמצא בטבלה עולה.

אני אממש את השימוש הראשון למסנן בלום (נמצא בעמוד 4, אופציה ראשונה), כאשר משתמשים חדשים ינסו להירשם לאתר, תבדק חוזק הסיסמא וכמו כן יהיה שימוש במסנן בלום לבדיקה האם שם המשתמש כבר קיים באתר, אם הוא קיים אז המערכת תבקש מהמשתמש להזין שם חדש.

ההסתברות לקבלת תשובה חיובית, כאשר למעשה האיבר אינו נמצא במערך:

ההסתברות שההסתברות שביט מסוים לא יודלק בידי אף אחת מפונקציות האש (יש k פונקציות כאלה) אחרי הכנסת n איברים, כאשר ההסתברות שביט מסוים לא יודלק בידי פונקציית האש, $1 - \frac{1}{m}$ (כאשר m זה גודל המערך):

$$\left(1 - \frac{1}{m}\right)^{kn}$$

ההסתברות שתתקבל תשובה חיובית עבור איבר חדש שאינו כבר נמצא במערך, תתרחש אם כל k הביטים שפונקציות הערבול נותנות עבורו כבר הודלקו, כאשר ההסתברות של ההסתברות שהערך

של ביט כלשהו אחרי הכנסת n איברים יהיה 1 היא $1 - \left(1 - \frac{1}{m}\right)^{kn}$ תהא:

$$\left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-\frac{kn}{m}}\right)^k$$

מהחישוב נובעים מספר מסקנות:

- (1) כלל שמספר התאים במערך גדל יש פחות סיכוי לקבלת תשובות חיוביות שגויות ולהיפך ככל שמוסיפים איברים כך הסיכוי עולה.
- (2) כשנבדוק איזה k לקבוע נראה שככל שהוא יותר גדול, ישנן יותר פונקציות האש ולכן יודלקו יותר ביטים כדי להחזיר תשובה שאיבר שייך לקבוצת איברים (כלומר הסיכוי לקבלת תשובה חיובית שגוייה קטן), אך כאשר מכניסים איבר למערך מדליקים עבורו יותר ביטים וגדלה ההסתברות שתהיה התנגשות בעתיד.

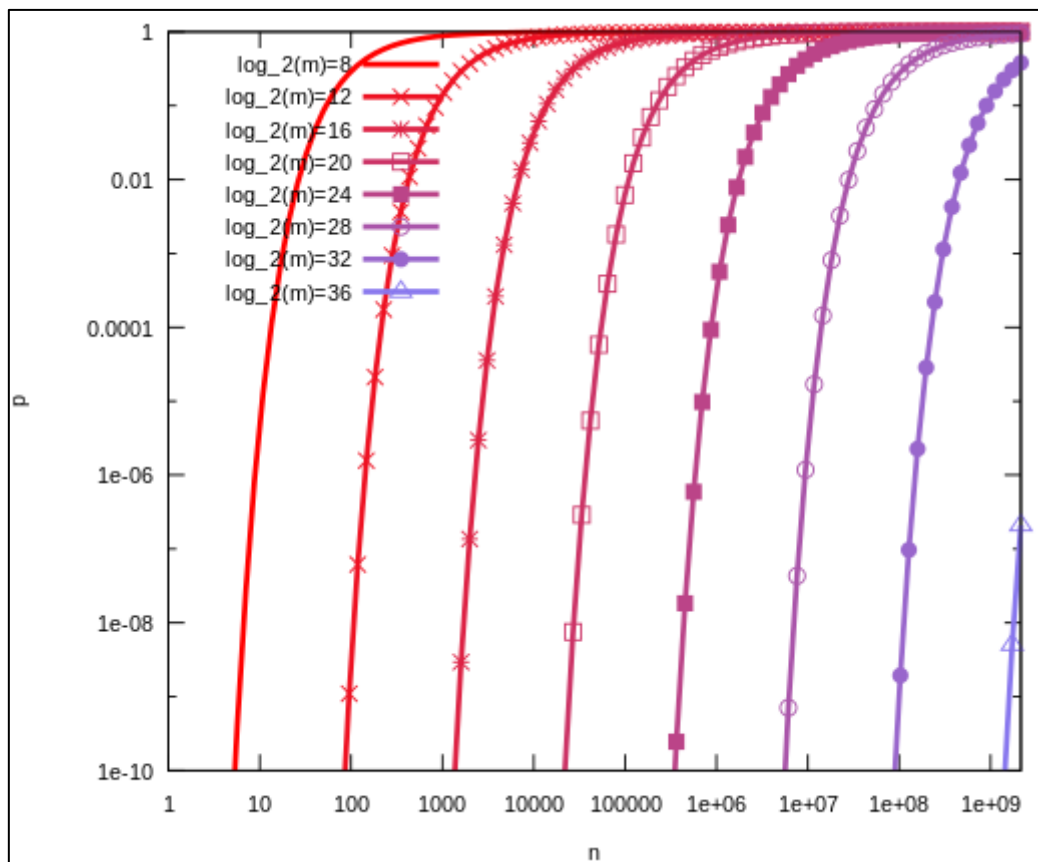
- (3) מספר האופטימלי של פונקציות האש הוא: $k = \frac{m}{n} \ln 2$ וההסתברות לטעות היא $\left(\frac{1}{2^{\ln 2}}\right)^{\frac{m}{n}}$.

ננסה להמחיש בצורה ויזואלית איך נראית ההסתברות לתשובות חיוביות שגויות, p , כפונקציה של גודל המערך ומספר האיברים בו, כאשר k יהיה:

$$p = \left(1 - e^{-\ln 2}\right)^{\frac{m \ln 2}{n}}$$

(התקבל מהצבה הערך האופטימלי של פונקצית האש בנוסחא הקודמת).

התוצאה התקבלה באיור (1) שבו ניתן לראות בציר y את ההסתברות p , בציר x את מספר האיברים במערך וכל שירטוט (צורה שונה לפי המקרא) מייצגת מערך שיש בו 2 בחזקת המספר רשום באותו בסיס.

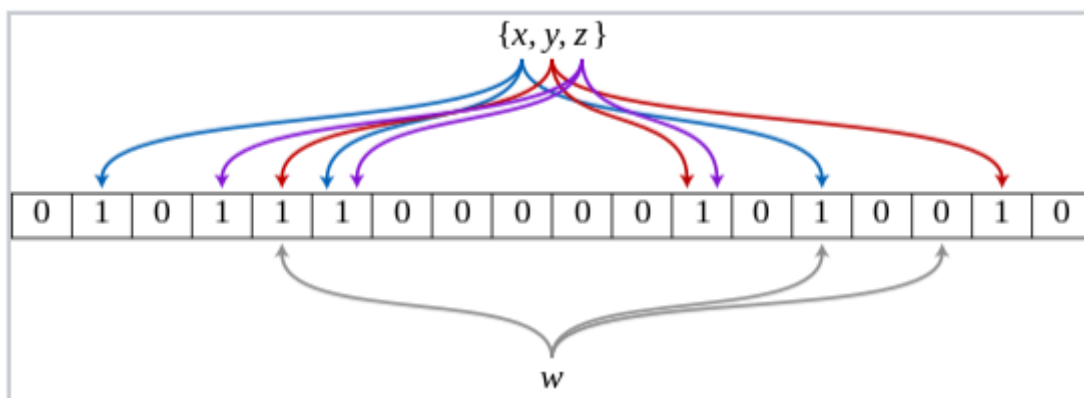


איור (1)- הסתברות לתשובה חיובית שגויה, p , כפונקציה של מספר האיברים, n , במסן בלום וגודל המסן, m , כאשר k נבחר להיות האופטימלי.

שימושים למסנן בלום:

- משתמשים במייל, כאשר משתמש רוצה ליצור חשבון חדש אז הוא מזין שם משתמש אשר ברצונו להירשם איתו אל השרת מיילים, כיוון שיכולים להיות המון משתמשים באותו שרת אז נעזרים במסנן בלום על מנת לצמצם את הזיכרון הנדרש כדי לשמור את השמות האפשריים הקיימים במערכת.
- Cache optimization - כאשר גולשים ברשת, המחשב שומר את URL לשם הורדת lag. אך ישנם פעמים שפונים אל אותו אתר פעם אחת בלבד, המונח נקרא "one-hit-wonder" ויצרוך זיכרון, אפשר להשתמש במסנן בלום שישמור כניסות לדפים אם נכנסו אליהם פעמים, כלומר לשמור את הדף בcache רק אם מסנן בלום החזיר תשובה חיובית
- המלצות ליוטיוב- בחלק העליון של הדף של אתר יוטיוב ישנן המלצות לסרטונים שעדיין לא נראו על ידי המשתמש. השרת מפעיל אלגוריתמים כדי למצוא סרטון בעל אותו תוכן הרלוונטי למשתמש ושולח אותו לדפדפן שלו. הדפדפן כולל מסנן בלום על ID של הוידאו. כל סרטון שנמצא שה-ID שלו כבר המצא במסנן בלום, תשובה חיובית, יוסר וכל תשובה חיובית תתווסף להצעות לסרטונים, כך תמיד יהיו למשתמש סרטונים חדשים להביט בהם.
- במסדי נתונים- oracle משתמשת במסנן בלום עבור:
 - הקטנת התקשורת בין תהליכים שרצים תחת התהליך הראשי בjoins מקבלי
 - לממש join-filter pruning: בגיזום מחיצות האופטימיזר מנתח משפטי SQL של WHERE- ו FROM להשמיט מחיצות שלא דרושות בזמן בניית רשימת הגישה למחיצות.
 - לסנן איברים בתאי exadata שונים: exadata מבצעים איחודים בין טבלאות גדולות וטבלאות lookup קטנות. על מנת לעשות זאת משתמשים במסנן בלום לקבוע האם שורה נמצאת כבר בסט התוצאה המבוקש.

לפני הצגת האלגוריתם, ניתנה דוגמה כדי לקבל אינטואיציה לגבי אופן פעולתו של המסנן. באיור (2) למטה, ניתן לראות דוגמה, למסנן בלום עם ערכים $m = 18, n = 3, k = 3$, כאשר מערך הביטים אשר מייצג רשימה של שלושה איברים שכבר קיימים במערכת: x, y, z ואיבר חדש w הנבדק האם הוא כבר קיים במערכת. לכל איבר מסומנים בעזרת חצים בצבעים שונים (כחול: x , אדום: y , סגול: z) הביטים אשר הוא מדליק במערך, כיוון שהאיבר w מסתכל על המקומות שמסומנים בעזרת החצים השחורים ובאחד במקומות הביט כבוי, אז תוחזר תשובה שהאיבר אינו נמצא במערך.



איור (2)- דוגמה למסנן בלום המייצג את הסט $\{x, y, z\}$. החצים הצבעוניים מציגים את המיקומים במערך הסיביות שכל אלמנט מוגדר ממופה אליו. האיבר w אינו נמצא בסט, כי אחד הביטים במקום שהוא ממופה אליו כבוי.

האלגוריתם עבור מסנן בלום:

מערך בינארי n ו- k פונקציות ערבול שונות, בעלות הטווח של גודל המערך. בדיקת אם איבר נמצא - מפעילים את פונקציית הערבול על המפתח של האיבר, אם k האינדקסים של ביטים במערך שקיבלנו הם בעלי ערך 1 אז יש סבירות שהאיבר כבר נמצא במערך, אם לפחות אחד מהמקומות בעל הערך 0 אז בוודאות האיבר אינו נמצא במערך. אם האיבר אינו נמצא נבצע: הכנסת איבר - מזינים את פונקציות ערבול בערכן (המפתח שלו) ומקבלים k אינדקסים של ביטים במערך אותן נדליק מ-0 ל-1.

האלגוריתם לחישוב חוזק הסיסמא יחושב לפי אתר password merter ומכיל את הכללים הבאים:

הדרישות המינימליות לסיסמא, היא שתהא מ 8 תווים המכילה 3 מתוך 4 הפריטים הבאים:

- אותיות גדולות
- אותיות קטנות
- מספרים
- סמלים

כאשר המילים באנגליות והציון הניתן לסיסמא מורכב מאפיונים המעלים את הערך הניתן לסיסמא (יותר בטוחה) או מקטינים את הערך (פחות בטוחה), כאשר הטווח הוא בין 0% ל-100%, והוספת עוד תווים אחרי סיסמא נבחרת כלשהי לא ישפר את התוצאה הסופית.

מאפיינים להעלאת ערך הסיסמא:

- מספר התווים – תוספת של n^4
- מספר האותיות הגדולות – תוספת של $2 * (len - n)$
- מספר האותיות הקטנות – תוספת של $2 * (len - n)$
- מספר תווים המספריים – תוספת של n^4
- מספר הסמלים – תוספת של n^6
- מספר המספרים האמצעים או התווים באמצע - n^2
- אם לפחות 3 מהדרישות המצויינות למעלה מתקיימות - n^2

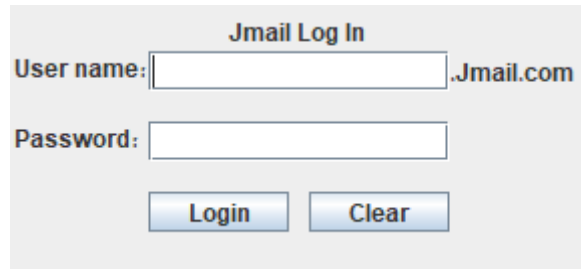
מאפיינים להורדת ערך הסיסמא:

- רק אותיות – n
- רק מספרים – n
- רצף של אותיות גדולות - $-(n^2)$
- רצף של אותיות קטנות - $-(n^2)$
- רצף של מספרים – $-(n^2)$
- רצף אותיות בסדר עולה או יורד, מעל 3 אותיות – $-(n^3)$
- רצף מספרים בסדר עולה או יורד, מעל 3 מספרים – $-(n^3)$
- רצף סמלים בסדר עולה או יורד, מעל 3 סמלים – $-(n^3)$

ממשק התוכנה ורכיבו:

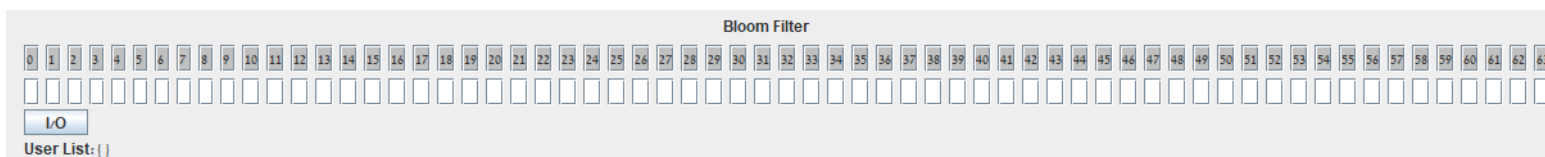
- **תצוגה ראשונית:**

הממשק מורכב משני חלקים, חלק של הרשמה הכולל שם משתמש וסיסמא(החלק השמאלי), המבוסס על המראה של מנגנון הרישום לGmail לפי מקור 10:



איור(3)- רישום משתמש חדש.

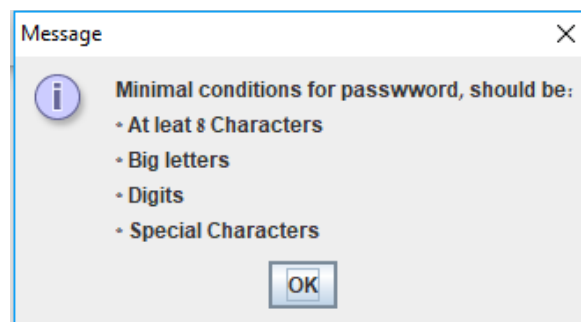
חלק שכולל את הייצוג של מסנן בלום ואת השמות הקיימים במערכת(החלק הימני):



איור(4)- ייצוג של מסנן בלום.

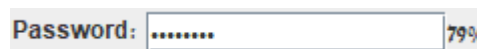
- **הכנסת סיסמא:**

הסיסמא צריכה לקיים את התנאים המינימליים לפני האלגוריתם של חוזק הסיסמא, אם הסיסמא אינה מסופקת לפי התנאים המינימליים, קופצת הודעת שגיאה.



איור(5)- הודעת שגיאה של סיסמא שלא מקיימת תנאים מינימליים.

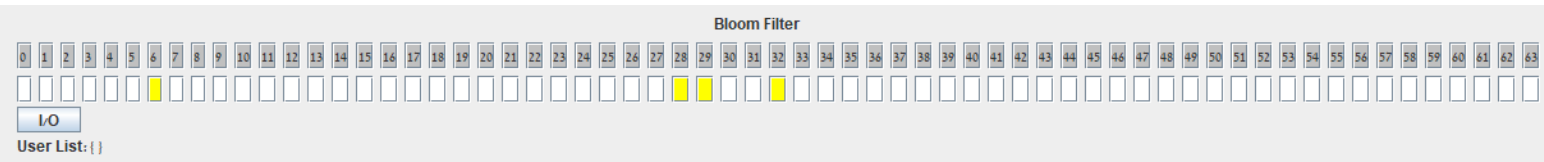
אחרי הכנסת סיסמא ולחיצה על enter או Login, ניתן לראות את חוזק הסיסמא מצד ימין למקום הכנסת הסיסמא. למשל עבור הסיסמא "1sdKL!@#":



איור(6)- הכנסת סיסמא והצגת החוזק שלה.

- **הכנסת שם משתמש:**

כאשר מכניסים שם משתמש ניתן ללחוץ, enter ואז לראות בסימון בצהוב את המקומות במסנן שהוא יתפוס, לפני הכנסתו לרשימת המשתמשים במערכת, למשל עבור שם במשתמש: "A"

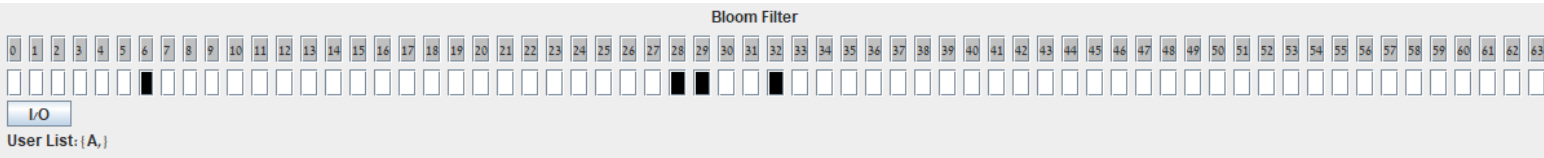


איור(7)- ייצוג משתמש "A", אחרי לחיצה של enter, לפני הכנסתו לרשימת המשתמשים במערכת.

ניתן להדליק ולכבות את המקומות שהוא יתפוס(בצהוב) ע"י כפתור ה **I/O**, כאשר מכניסים שם משתמש אז צריך ללחוץ enter, כדי שכפתור ה **I/O** יתעדכן, אך אם מכניסים שם משתמש אך לא מבצעים enter אז כפתור ה **I/O** יכיל את שם המשתמש הקודם, כיוון שהאפליקציה לא יודעת שהוזן שם משתמש.

- **ביצוע Login:**

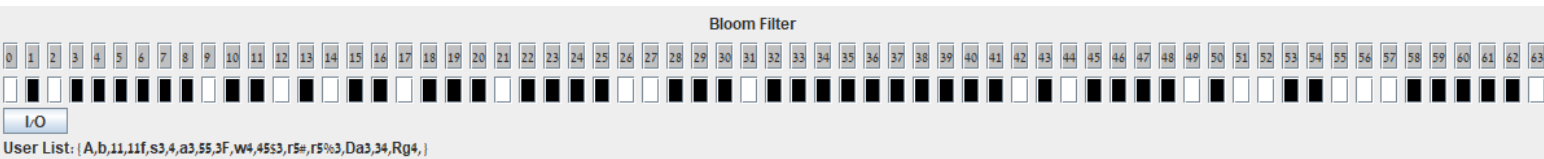
כאשר מבצעים login אז מכניסים את השם משתמש למערכת, ניתן לראות אותו ב user list, כמו כן צובעים בשחור את המקומות של הייצוג במסנן בלום, למשל עבור שם המשתמש "A",



איור(8)- הכנסת משתמש "A" למערכת.

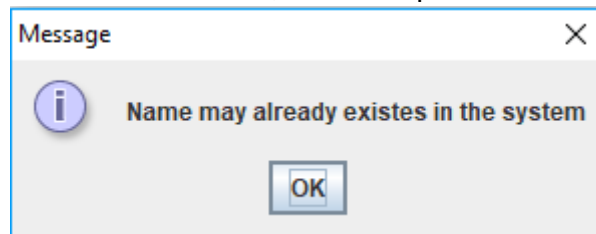
- **התנגשות במקומות במסנן:**

לפי הדוגמה ראוי לציין שכאשר מכניסים שם משתמש חדש, הסיסמה הקודמת אינה נמחקת משורת הטקסט מטעמי נוחות, כאשר רוצים להכניס הרבה משתמשים ולראות את פעולת המסנן ביתר נוחות. דוגמה להכנסה של מספר משתמשים למערכת:



איור(9)- דוגמה להכנסה של מספר משתמשים למערכת.

כעת ננסה להכניס משתמש שכבר קיים במערכת, למשל את שם במשתמש "A":



איור(10)- דוגמה להכנסה המשתמש שכבר קיים במערכת, "A".

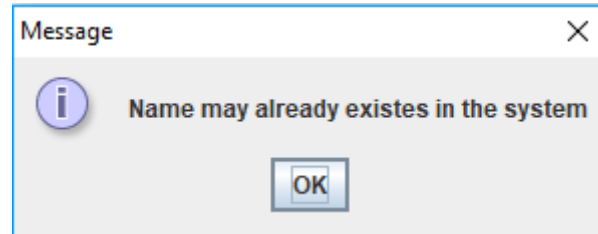
כעת ננסה להכניס את המשתמש הבא, "AaBbCcDdEe":

Jmail Log In

User name: .Jmail.com

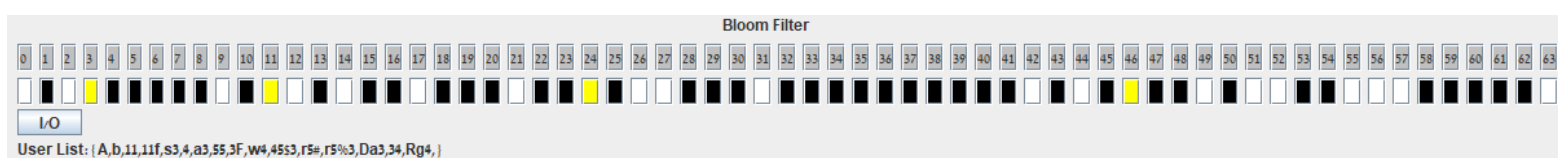
איור(11)- הכנסת משתמש "AaBbCcDdEe" למערכת.

ונקבל את ההודעה הבאה:



איור(12)- דוגמה להכנסה המשתמש "AaBbCcDdEe", שאינו קיים במערכת.

אם נרצה לראות את המקומות ששם המשתמש, "AaBbCcDdEe" תופס, כעת נוכל ללחוץ על הכפתור ה-10 כדי לכבות ולהדליק את המקומות שהמשתמש מנסה לתפוס:



איור(13)- מקומות ששם המשתמש "AaBbCcDdEe", מנסה לתפוס במערכת.

ניתן לראות מאיור (9) שהמקומות 3,11,24,46 כבר תפוסים.

- (1) <https://www.quora.com/What-are-the-best-applications-of-Bloom-filters>
- (2) <https://kunigami.blog/2015/01/29/bloom-filters/>
- (3) <https://www.geeksforgeeks.org/bloom-filters-introduction-and-python-implementation/>
- (4) <https://juliandontcheff.wordpress.com/2012/08/28/bloom-filters-for-dbas/>
- (5) <http://www.passwordmeter.com/>
- (6) https://en.wikipedia.org/wiki/Bloom_filter
- (7) <http://lilmlib.github.io/bloomfilter-tutorial/>
- (8) https://www.javamex.com/tutorials/collections/bloom_filter_java.shtml
- (9) <https://asecuritysite.com/encryption/bloom>
- (10) <https://accounts.google.com/signup/v2/webcreateaccount?hl=en-GB&flowName=GlifWebSignIn&flowEntry=SignUp>