ממן 14:

סעיף א:

בהתחלה אנחנו מכניסים את האיברים למבנה וכל הכנסה של איבר למבנה אנחנו מדליקים את הביטים שלו בטבלת הביטים.בביצוע הבדיקה הביטים של איבר שייך למבנה כבר מודלקים מעצם העובדה שהוא נכנס למערך מהתחלה.מכאן נובע שהסיכוי שאיבר שייך למבנה יוכרז כלא שייך הוא אפס

סעיף ב:

נניח שהפונקציות הגיבוב הן בעלות התפלגות אחידה. ההסתברות שביט מסוים לא יודלק בידי פונקציית גיבוב היא 1-1/m

ההסתברות שביט מסוים לא יודלק בידי אף אחת מפונקציות הגיבוב היא $(1-1/m)^k$

ההסתברות ש- $h_i(a)=0$ אחרי הכנסת n איברים היא $(1-1/m)^{kn}$

ועל כן ההסתברות $h_i(a)=1$ לאחר הכנסת n איברים היא $1-(1-1/m)^k$

הסיכוי שאיבר לא שייך למבנה יוכרז כשייך הוא $(1-(1-1/m)^{kn})^k$

דבר זה יקרה אם כל k הביטים שפונקציות הגיבוב נותנות עבורו הודלקו.

סעיף ג:

$$(1-(1-1/32*10^6)^{13*10^6})^{13}=0.00000006401416$$

סעיף ד:

מטרת הפרויקט:

מטרת הפרויקט לבנות מערכת אשר תוכל לבדוק במהירות שייכות של איבר למבנה נתונים

תיאור מבנה הנתונים-Mytable:

נשתמש בטבלת גיבוב המכילה טבלת ביטים בגודל m וseeds פונקציות גיבוב(hash)המחזירות כל אחת ערכים בין 0 ל- m-1 שכל ערך הוא המיקום בטבלת הביטים של פונקציה על איבר

בניה Mytable:

- נגדיר מערך מסוג int שמקבלת טבלת ביטים בגודל קבוע- O(1)
- נגדיר מספר הפונקציות seeds בגודל קבוע- O(1)

סה"כ זמן ריצה : O(1)

פעולות:

הוספת מספר insertElement(int num):

הוספת מספר למבנה נתונים

עבור כל מספר שנכנס למבנה נתונים אנחנו נעשה עליו לולאה שמספרה הוא מספר הפונקציות שהגדרנו(seeds)שתעשה את הפעולות הבאות(O(1) לולאה עם מספר קבוע):

- נשתמש בפונקציה MurmurHash3 על מספר(a),זמן הריצה O(n)
- אחרי זה נעשה ערך מוחלט על מספר(a) ונחלק אותו בגודל המערך של הביטים (על מנת שהערך הזה לא יחרוג מגבולות המערך הביטים),זמן ריצה O(1)
- נכנס לתא במיקום הערך(a) את המספר 1 בטבלת הביטים,זמן ריצה O(1)

סה"כ זמן ריצה : O(n)

הוספת מילה insertElement(string word):

הוספת מילה למבנה נתונים

נחשב כל תו של המילה ע"י טבלת אסקי נסכום את התווים(sum), (לולאה שעוברת על כל התווים במילה ולכן הלולאה מתבצעת כאורך המילה שזה O(n))

עבור כל מספר שנכנס למבנה נתונים אנחנו נעשה עליו לולאה שמספרה הוא מספר הפונקציות שהגדרנו(seeds)שתעשה את הפעולות הבאות((O(1 לולאה עם מספר קבוע):

- נשתמש בפונקציה MurmurHash3 על מספר(a),זמן הריצה (O(n
- אחרי זה נעשה ערך מוחלט על מספר(a) ונחלק אותו בגודל המערך של הביטים (על מנת שהערך הזה לא יחרוג מגבולות המערך הביטים),זמן ריצה (O(1
- נכנס לתא במיקום הערך(a) את המספר 1 בטבלת הביטים,זמן ריצה (O(1

סה"כ זמן ריצה : (O(n

בדיקת מספר-checkElement(Integer num):

בדיקת שייכות של מספר במבנה נתונים ע"י טבלת ביטים

עבור כל מספר שנכנס למבנה נתונים אנחנו נעשה עליו לולאה שמספרה הוא מספר הפונקציות שהגדרנו(seeds)שתעשה את הפעולות הבאות((O(1 לולאה עם מספר קבוע):

- נשתמש בפונקציה MurmurHash3 על מספר(a),זמן הריצה (O(n
- אחרי זה נעשה ערך מוחלט על מספר(a) ונחלק אותו בגודל המערך של הביטים (על מנת שהערך הזה לא יחרוג מגבולות המערך הביטים),זמן ריצה (O(1
- אם הערך שחישבנו(a) שהוא המיקום בטבלת הביטים שווה למספר 1 אז תוסיף אחד למספר true שיש למספר בטבלת הביטים(count),זמן ריצה (O(1

לאחר הלולאה,אם count שווה למספר הפונקציות(seeds) תחזיר true אחרת תחזיר false,זמן ריצה (O(1

סה"כ זמן ריצה : (O(n

בדיקת מילה-checkElement(string word):

בדיקת שייכות של מילה במבנה נתונים ע"י טבלת ביטים

נחשב כל תו של המילה ע"י טבלת אסקי נסכום את התווים(sum), (לולאה שעוברת על כל התווים במילה ולכן הלולאה מתבצעת כאורך המילה שזה (O(n

עבור כל מספר שנכנס למבנה נתונים אנחנו נעשה עליו לולאה שמספרה הוא מספר הפונקציות שהגדרנו(seeds)שתעשה את הפעולות הבאות((O(1 לולאה עם מספר קבוע):

- נשתמש בפונקציה MurmurHash3 על מספר(a),זמן הריצה (O(n
- אחרי זה נעשה ערך מוחלט על מספר(a) ונחלק אותו בגודל המערך של הביטים (על מנת שהערך הזה לא יחרוג מגבולות המערך הביטים),זמן ריצה (O(1
- אם הערך שחישבנו(a) שהוא המיקום בטבלת הביטים שווה למספר 1 אז תוסיף אחד למספר true שיש למספר בטבלת הביטים(count), זמן ריצה (O(1

לאחר הלולאה,אם count שווה למספר הפונקציות(seeds) תחזיר true אחרת תחזיר false זמן ריצה (O(1

סה"כ זמן ריצה : (O(n

Example:

הרעיון אלגוריתם:

בודקת במהירות שייכות של איבר למבנה נתונים

אלגוריתם מקבל שתי קבצים מסוג string:(שניהם (O(1

- קובץ המכיל איברים מספרים או מחרוזות(מופרדים בפסיקים)של איברים להכנסה למבנה (insert.txt)
- קובץ המכיל איברים מספרים או מחרוזות(מופרדים בפסיקים)של איברים לבדיקת שייכות למבנה(check.txt)

אחרי זה אלגוריתם מקבל גודל המערך(m) ומספר פונקציות גיבוב (k) מהמשתמש ויוצר טבלת גיבוב מהנתונים הללו ומהפונקציה Mytable זמן ריצה (O(1

אחרי זה אם הקובץ האיברים להכנסה למבנה קיים אז הקריאה נעשית ע"י פונקציית Scanner ומשתנה Token שהן לוקחות איבר מקובץ ללא פסיקים,לולאה שעוברת על כל המילים בקובץ ולכן הלולאה מתבצעת כמספר המילים בקובץ שזה O(n)
אחרי זה בודקים אם האיבר הוא מילה או מספר:
אם זה מספר אז מפעילים עליו את הפונקציה **insertElement(int num)**,זמן ריצה O(n)
אם זה מילה אז מפעילים עליו את הפונקציה **insertElement(string word)**,זמן ריצה O(n)
אחרת מדפיס שהקובץ לא קיים

אחרי זה אם הקובץ האיברים להכנסה למבנה קיים אז הקריאה נעשית ע"י פונקציית Scanner ומשתנה Token שהן לוקחות איבר מקובץ ללא פסיקים,לולאה שעוברת על כל המילים בקובץ ולכן הלולאה מתבצעת כמספר המילים בקובץ שזה O(n)
אחרי זה בודקים אם האיבר הוא מילה או מספר:
אם זה מספר אז מפעילים עליו את הפונקציה **checkElement(Integer num)**-ואז מדפיסה על המסך אם המספר מופיע במבנה או לא (זמן ריצה O(n))
אם זה מילה אז מפעילים עליו את הפונקציה **checkElement(string word)**-ואז מדפיסה על המסך אם המילה מופיע במבנה או לא (זמן ריצה O(n))
אחרת מדפיס שהקובץ לא קיים

סה"כ זמן ריצה : $O(n^2)$