**מיכאל 316153865**

**אביב ששון: 318305034**

**תרגיל 4 - פתרון:**

**מבנה הנתונים מכיל:**

1. **Countries-** מערך בגודל N +1 (התא הראשון יהיה ריק). האיבר ה- i במערך מצביע לאובייקט המכיל מבנה נתונים - cal\_day,Di,days כאשר

* **Di -** מספר הימים עבור מחזור הבחירות
* **- cal\_day** מצביע למערך D שמייצג את הלוח ימים של הבחירות (סקאלת ימים מהיום ה-0 עד יום D).
* **days -** מספר הימים שנותרו כמתואר.

1. **Calendar-** מערך בגודל D המכיל מספרים בסדר עולה עד למספר D (כמתואר בתרגיל).

**מימושים לפונקציות:**

1. **Init(D,N) -** ניצור מערך בגודל N+1 עבור המדינות, ומערך בגודל D עבור סקאלת הימים מ- 0 עד D.
2. **Insert(i,Di,days) –** נאתחל את המדינה ה- i למקומה, בנוסף גם את השדות שלה - days ו- DI, Cal\_day . Cal\_day יצביע ליום במערך D המתקבל מהפרש  
    Di – days. אם המדינה כבר אותחלה או שDi > D , תוחזר שגיאה בהתאם.
3. **ElectionsToday() -** נעבור כל המדינות שאותחלו במערך Countries.  
   עבור כל מדינה נקדם את המצביעcal\_day ליום הבא (בהתאמה days מתעדכן). במידה ו-cal\_day מצביע לתא שערכו שווה ל Di, נדפיס את המדינה ה-i ונעדכן את cal\_day להצביע לראש מערך D.
4. **PredateElections(I, days) -** נקדם אתcal\_day ב- days ימים , במידה וdays- גדול ממספר הימים שנותרו תוחזר שגיאה בהתאם.  
   במידה ויום הבחירות הגיע לאחר הקידום נוסיף על כך הערה (?).

**נכונות:**

1. Init(N,D) - מאתחל את מבנה הנתונים עם הפרמטרים N ו- D כדרוש.
2. Insert(I,Di,days) – מוסיף את המדינה הi במידה והיא לא כבר במבנה, ושומר את הקבוע Di בתוך האובייקט של המדינה.
3. ElectionsToday() – מדפיס את האינדקסים של המדינות בהם תכנו של המצביע cal\_day שווה לDi .
4. PredateElection(I,days) – בהינתן המדינה ה-I נוכל לחשב את מספר הימים שנותרו לבחירות לאחר ההקדמה על ידי כך שנקדם את המצביע cal\_day days, פעמים קדימה .  
   במידה והזזנו את המצביע מעבר ליום הבחירות המיועד (Di ) אזי זאת תהיה שגיאה כיון שהוא כבר מתבצע קודם לכן ואין טעם להקדימו יותר ימים ממה שהוא מתוכנן.

**סיבוכיות:**

1. Init(N,D) – אתחול שני מערכים בגודל D ו N+1 כפי שלמד בכיתה בסיבוכיות O(1).
2. Insert(I, Di, days) – גישה למערך, אתחול של מערך ואתחול שלושת הנתונים בסיבוכיות O(1).
3. ElectionsToday() – כפי שנלמד בכיתה נעבור על המערך של קיצורי הדרך , כלומר על המדינות שאותחלו בלבד ומשם נגיע לאינדקס שלהם במערך , בדיקת התנאים על כל מדינה נעשית ב-O(1) ולכן כאשר k מספר המדינות שאותחלו אז הסיבוכיות היא O(k).
4. PredateElection(I, days) – גישה למסםר קבוע של איברים במערך והזזת מצביע במערך בסיבוכיות O(1).
5. סיבוכיות מקום: הקצינו מערך בגודל N+1 ובגודל D ולכן סיבוכיות המקום היא O(N + D) .

**תרגיל 5 - פתרון:**

**מבנה הנתונים מכיל:**

Countries – מערך בגודל של 52 שהוא ייצג את כל המדינות וכל תא יצביא למערך בגודל N (ages\_group ) שהוא מייצג את קבוצות הגיל השונות

Ages\_group – 52 מערכים בגודל N שמייצג את קבוצות הגיל וכל תא יכיל את מספר החיסונים שנותרו למדינה מסוימת ולקבוצת הגיל הנוכחית.

C – קבוע שיכיל את מספר החיסונים ההתחלתי שכל מדינה מקבל לכל קבוצת גיל.

Sum- משתנה שמייצג את מספר החיסונים שהתבצעו בכל ארה"ב היום, מאותחל ל0.

Priority\_queue – רשימה מקושרת שבה כל אובייקט מייצג מספר מסוים של כמות מתחסנים (בסדר עולה החל מ1) לכל קבוצת גיל כלשהי.  
כל צומת i ברשימה המקושרת מצביעה לרשימה מקושרת "list\_ages<i>" כאשר בה נמצאים קבוצות הגילאים שלהם יש את מספר מתחסנים i ,בכל המדינות.  
  
בכל פעם שנגיע למספר מתחסנים שעדיין לא קיים ברשימה , כלומר גדול יותר מהמספר המקסימלי שהיה, ניצור אובייקט חדש שמייצג את מספר זה ונכניס אותו לראש הרשימה. כך שנקבל שבראש הרשימה יש את קבוצות הגיל עם הכי הרבה מתחסנים עד כה.

**Ages\_info –** מערך בגודל n, כך שכל תא מייצג מידע על קבוצת גיל i בכל ארצות הברית. בכל תא במערך זה קיים אובייקט המכיל בתוכו 3 שדות – הראשון הוא סכום החיסונים שנעשו עד כה, השני הוא מצביע לnode ברשימה המקושרת Priority\_queue (שמייצג כמות מתחסנים) והשלישי הוא מצביע לNode ב - list\_ages<i> שמייצג את קבוצת הגיל i.

**מימושים לפונקציות:**

Init(n,c) – ניצור מערך דו מימדי n x 52 שייצג את מספר המדינות ולכל מדינה ייצג את n קבוצו’ הגילאים שמקבלים את החיסון (countries , ages\_group ).  
בנוסף ניצור את האובייקט שמכיל שלושה שדות (סכום חיסונים, מצביע לnode בpq , מצביע לnode ב list\_ages<i>). וניצור מערך של האובייקטים האלה בשם Ages\_info

Vaccinate(I, k) – ניגש למדינה ה k ולקבוצת הגיל I

* אם K לא בטווח 1-52 אזי תוחזר שגיאה
* אם המדינה וקבוצת הגיל אותחלו והמערך במקום הנ"ל שווה ל0 , כלומר לא נותרו חיסונים נחזיר הודעה " finished" .
* אם המדינה וקבוצת הגיל אותחלו (arr[k][i[) אזי נחסר 1 מהערך של המערך במקום הנ"ל
* אם עדיין לא אותחלו אזי נאתחל אותם לערך c-1 בהתאם לקבוע שנמצא במבנה.
* מעדכן את sum (sum++).
* בכל פעולת עדכון של כמות המתחסנים לקבוצת גיל מסוימת (בין אם היא אותחלה או לא) מתבצעות מספר פעולות עדכון במערך המידע של הגילאים.
* נוסף "אחד" לסכום המתחסנים של אותה קבוצת גיל.
* בהתאמה, נקדם את קבוצת הגיל בpq ול list\_ages<i+1>, אם לא קיימת כזו, ניצור node נוסף מתאים ונשבץ אותו בראש הרשימה PQ, ונעדכן את המצביעים ב **Ages\_info** כך שנוכל לגשת לאיבר הנ"ל ללא תלות בn.

Add(I, k) - ניגש למדינה ה k ולקבוצת הגיל I

* אם K לא בטווח 1-52 אזי תוחזר שגיאה.
* אם קבוצת הגיל הI במדינה הK אותחלה אזי נוסיף 1+ לתא במערך במקום המתאים.
* אם עדיין לא אותחלו אזי נאתחל אותם לערך c+1 בהתאם לקבוע שנמצא במבנה.

Vaccinated()- נחזיר את sum שנמצא במבנה.

Popular(m) – נעבור על הרשימה המקושרת priority\_queue מהתחלה.

לכל צומת ברשימה "נכנס" לרשימת list\_ages<i> ונמנה לכל היותר m (בעזרת counter) ,קבוצות גיל כך שלעל קבוצת גיל שנמנית היא תודפס בהתאם.  
עבור מספר קבוצות גיל ב list\_ages<i> שקטן מ- m נעבור לצומת הבאה ונמשיך למנות ולהדפיס את הקבוצות עד שנגיע לm.  
עבור m גדול ממספר קבוצות הגיל המתחסנים בכל ארצות הברית, נדפיס את מספר הקבוצות שלהם יש מתחסנים עד כה.  
כיון שרשימה זו מונה את מספר המתחסנים מהגבוה לנמוך לכן נבטיח שm הקבוצות גיל הראשונות הם הגבוהות ביותר מבחינת מספר מתחסנים

**נכונות:**

1. Init(n, c) – מאתחל את מבני הנתונים עם הפרמטרים n, c כנדרש.
2. Vaccinate(I, k) – מחסר 1 מהמקום המתאים במערך הדו-ממדי למדינה ולקבוצת הגיל המתאימה. במידה ונגמרו החיסונים, תוחזר הודאה בהתאם.  
   מעדכן את קבוצת הגיל המתאימה בages\_info , מקדם את קבוצת הגיל ברשימת pq כולל עדכון המצביעים בהתאם בpq וגם ב ages\_info.
3. Add(I, k) – מוסיף 1 במקום המתאים למדינה ולקבוצת הגיל המתאימה כך שיש להם עוד חיסון נוסף במערכת של המדינה הi לקבוצה הk.
4. Vaccinated()- מחזיר את הערך sum שעלה ב1 בכל פעם שמישהו התחסן.
5. Popular(m) – מחזיר את m קבוצות הגיל שהתחסנו הכי הרבה חיסונים מבין שאר קבוצות הגיל ע"י ריצה על רשימת הרשימות pq.

**סיבוכיות:**

1. Init(n, c) – אתחול מערך בגודל 52X n , מערך בגודל מספר קבוצות הגיל (n), שני משתנים ורשימת רשימות בסיבוכיות O(1) כפי שנלמד בכיתה.
2. Vaccinate(I, k) – גישה למערך פעולות על מערך והזזת איברים ברשימה מקושרת תוך כדי שיודעים מראש את הכתובת שלהם בסיבוכיות O(1)..
3. Add(I, k) – גישה למערך תוך בדיקת תנאים בסיבוכיות O(1).
4. Vaccinated()- גישה למשתנה במבנה הנתונים בסיבוכיות O(1).
5. Popular(m) – רץ על m המדינות הראשונות ברשימת רשימות pq כאשר הריצה ברשימה הפנימית הראשונה וכן הלאה עד שהמונה מגיע לm בסיבוכיות O(m).