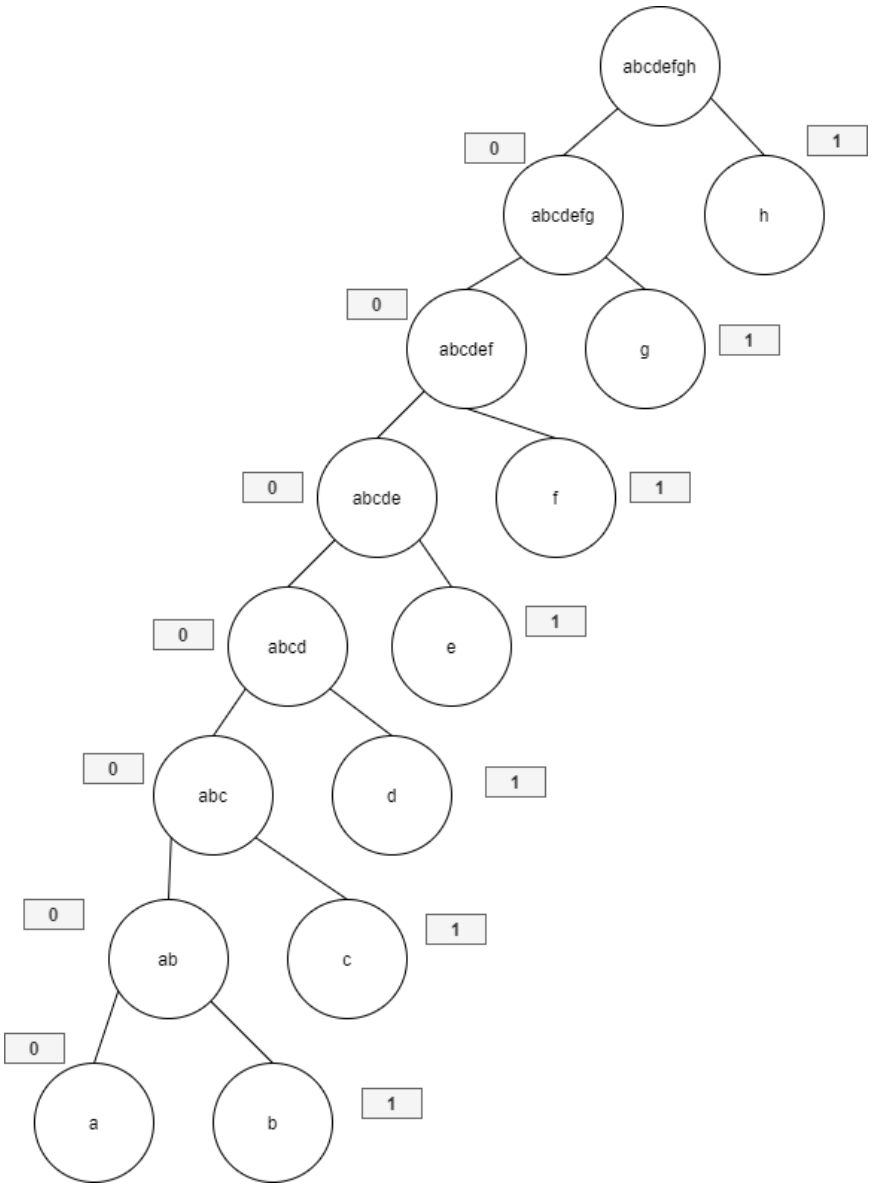
תרגיל בית מס' 6 במבוא מורחב למדעי המחשב – 315328963

שאלה 2

א. להלן עץ הקידוד עבור הקורפוס:



והקוד הוא: $\{h: 1, g: 01, f: 001, e: 0001, d: 00001, c: 000001, b: 0000001, a: 0000000\}$.

ב. נשים לב כי עבור כל מספר פיבונאצ'י i, מתקיים כי מס' פיבונאצ'י הקודם (כמובן) והסכום של כל מספרי פיבונאצ'י עד אליו (עד הקודם) קטנים ממספר הפיבונאצ'י i. כלומר נקבל עץ שבו מספר הפיבונאצ'י הגדול ביותר נמצא קרוב ביותר לצמרת.

מספר הפיבונאצ'י שבא לפניו (והוא 'הבא להיסכם'), יהיה גם מצד ימין, רק 'מדף' אחד נמוך יותר.
בתחתית יהיו 2 מספרי פיבונאצ'י בעלי אותו מספר ביטים, כלומר בעלי n-1 ביטים, כאשר לימני יהיה אפסים ובקיצוני מימין אחד, ולשמאלי יהיו רק אפסים. נשים לב שמכיוון שמספר פיבונאצ'י הראשון והשני שווים, ניתן להחליף ביניהם
כלומר בהכללה:

$$\underline{bits} = \begin{cases} ,'0' * (n-1), & n = 1 \ \boldsymbol{or} \ n = 2 \\ '0' * (n-2) + '1', & n = 1 \ \boldsymbol{or} \ n = 2 \\ '0' * (n-i) + '1', & else, i \ is \ indexof \ Fib \ number \end{cases}$$

ג. נראה כי ההפרש ביניהם הוא אפס כי מתקיים $|C(a_1)| = |C(a_n)|$.
נתחיל לבנות את קוד האפמן. נשים לב כי שני המינימליים הם a_1, a_2, אז 'נחבר' אותם ובשורש שלהם נקבל את המספר $a_1 + a_2$ שמקיים לפי הנתונים: $a_1 + a_2 > a_1 + a_1 = 2a_1 > a_n$. אם כן, כעת המספר המקסימלי ברשימה הוא $a_1 + a_2$, ולכן המספרים המינימליים יהיו a_3, a_4, אז 'נחבר' אותם ובשורש שלהם נקבל את המספר $a_3 + a_4$ שמקיים לפי הנתונים: $a_3 + a_4 > a_1 + a_2$. כלומר, כעת המספר $a_3 + a_4$ הוא המקסימלי מבין כל המספרים שנותרו.
נמשיך את התהליך בצורה רקורסיבית, ולבסוף נחבר את a_{n-1}, a_n ונקבל כי המספר $a_{n-1} + a_n$ יהיה כעת המקסימלי מבין כל הזוגות המחוברים. לכן הזוגות המינימליים כעת הם a_1, a_2 ו-a_3, a_4.
נוכל לחבר אותם ולהמשיך עם התהליך באופן רקורסיבי, עד שיהיו לנו שני מספרים, $a_1 + a_2 + \cdots + a_{\frac{n}{2}}$, $a_{\frac{n}{2}+1} + \cdots . + a_{n-1} + a_n$ אותם נחבר ונקבל את השורש של עץ האפמן.
נשים לב, כי בסה"כ בוצעו פעולות של חיבור לשורש $\log(n)$ פעמים, כאשר גם a_1 וגם a_n עברו $\log(n)$ פעולות חיבור כאלה. כלומר העומק של כל אחד מהם בעץ ההאפמן שנוצר יהיה זהה, $\log(n)$, ולכן מס' הביטים בקידוד שלהם, שנקבע לפי עומק העץ עד האיבר, יהיה גם זהה: $|C(a_1)| = |C(a_n)| = \log(n)$.

שאלה 3

ב. סיבוכיות הזיכרון היא $O(Nlog(N))$. כפי שנאמר בשאלה, שמירה של כל אינדקס במחרוזת באורך N לוקחת $\log(N)$ ביטים. נשים לב כי יש $N-2=O(N)$ אינדקסים לשמור (כל אינדקס מייצג תת-מחרוזת באורך 3), ולכן בסה"כ סיבוכיות הזיכרון תהיה $O(N)\cdot O(\log(N))=O(Nlog(N))$.

ג. סיבוכיות זמן הריצה היא $O(\max\_length)$. נסתכל על מימוש הפונקציה. הפונקציה מבצעת פעולות אריתמטיות פשוטות (סיבוכיות זמן קבועה), ולאחר מכן פעולת חיפוש במילון – בממוצע בסיבוכיות קבועה. על כל 'התאמה' למחרוזת 3 התווים, הפונקציה מוסיפה אותו לרשימה אחרת. פעולה כזאת היא בסיבוכיות זמן קבועה, וישנו מספר קבוע של התאמות כאלה, אז הסיבוכיות של לולאה זאת היא סיבוכיות זמן קבועה $O(1)$.

לאחר מכן הפונקציה נכנסת עבור כל האתמה (ויש מספר קבוע שלהן) ללולאת while, שרצה כל עוד k קטן מ-max_length. כלומר, הפונקציה תרוץ במקרה הגרוע $O(\max\_length)$ פעמים. הפעולות בתוך הלולאה הן פעולות אריתמטיות ופעולות השמה, שהן בסיבוכיות זמן קבועה. כלומר זמן הריצה של הלולאה יהיה $O(\max\_length)$.

נסכום את את הסיבוכיות של הלולאות ונקבל $O(1)+O(maxlength)=O(\max\_length)$.

שאלה 4

א. מרחק Hamming של הקוד (המינימלי) הוא 2. נניח למשל שישנו שוני בביט אחד מביטי ה'אינפורמציה', אז הדבר יגרום לשינוי בביטי ה'ייצוג', כאשר שינוי מינימלי כזה יהיה של ביט אחד. ניתן דוגמה להמחשה של מרחק מינימלי: 1111111000, והקרוב אליו ביותר (בין היתר) יהיה 1011111001, ניתן לראות כי המרחק ביניהם הוא 2.

ב. i. כפי שלמדנו בכיתה, כאשר d הוא המרחק המינימלי של הקוד, d-1 יהיה מספר השגיאות שניתן לגלות, ולכן במקרה זה מספר השגיאות שניתן יהיה לגלות הוא 1.

ii. כפי שלמדנו בכיתה, כאשר d הוא המרחק המינימלי של הקוד, $\left\lfloor\frac{d-1}{2}\right\rfloor$ יהיה מספר השגיאות שניתן לתקן, ולכן במקרה זה מספר השגיאות שניתן יהיה לתקן הוא 0 (כלומר לא ניתן לתקן).

ג. במצב כזה נוכל תמיד לגלות כל כמות של שגיאות. זאת משום שצריכה להיות התאמה בין שלושת הביטים מימין לבין 7 הביטים. בשל הכלל שניתן לנו, נוכל לדעת האם המספר שנוצר בשלושת הביטים מימין אכן שווה למספר האפסים בשבעת הביטים שמשמאל.

למשל, עבור 1100111011, אנו רואים כי ישנם במספר משמאל 2 אפסים, ואילו הביטים שמימין מייצגים את המס' 3. אז נוכל להסיק כי ישנה שגיאה אחת – האפס הקיצוני הימני הפך ל-1.

עבור 1100111111, אנו רואים כי ישנם במספר משמאל 2 אפסים, ואילו הביטים שמימין מייצגים את המס' 7. כלומר נוכל להסיק כי ישנן 2 שגיאות – האפס הקיצוני הימני והאפס השלישי מימין הפכו שניהם לאחד. לחילופין – במספר השמאלי אחד האפסים הפך לאחד, ואילו בימני, האפס השלישי מימין הפך לאחד.

דוגמה נוספת, עובר 1111111111, אנו רואים כי ישנם במספר משמאל 0 אפסים, ואילו הביטים שמימין מייצגים את המס' 7. לכן נוכל להסיק כי ישנן 3 שגיאות – שלושת האפסים של המספר השמאלי (מייצגים את המס' 0) הפכו לאחדים.

וכן הלאה.

שאלה 5

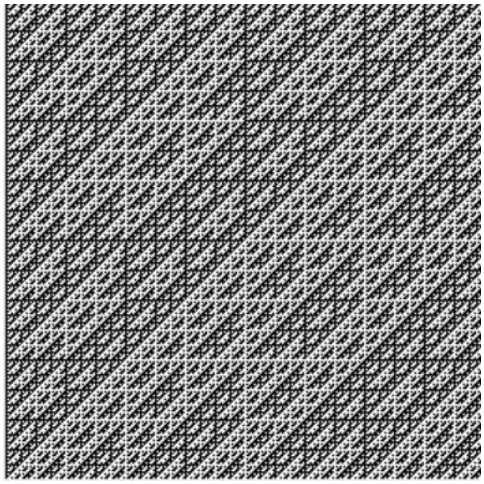
א. ניתן ליצור פונקציה שכזאת. הפונקציה תריץ את האיבר הראשון של הגנרטור הראשון, ולאחר מכן תשמור את האיבר הזה בזכרון עזר. הפונקציה תבדוק מהו האיבר הבא בגנרטור השני, ואם הוא לא נמצא בזכרון העזר, היא תחזיר אותו. אם כן נמצא בזיכרון העזר, הפונקציה לא תחזיר אותו אלא תעבור לבדיקת האיבר הבא בגנרטור הראשון. כלומר, באופן כללי, אם איבר לא נמצא בזכרון העזר, משמע הוא עדיין לא הוחזר, ולכן הפונקציה תחזיר אותו, ואם הוא כן, אז הוא כבר הוחזר ולא יכול להיות מוחזר עוד פעם, לכן הפונקציה תפסח עליו.

ב. לא ניתן ליצור פונקציה כזאת. כאן, על מנת שהפונקציה תחזיר איבר, היא צריכה קודם לוודא כי שני הגנרטורים אכן מחזירים את המספר. לכן, כאשר הגנרטור הראשון מחזיר מספר, היא תצטרך לשמור אותו בזכרון עזר ולחכות שגם הגנרטור השני יחזיר את המספר הזה (ורק אז להחזיר אותו). דבר זה עלול לקחת זמן רב, כלומר ייתכן ויעבור זמן רב, אפילו אינסופי מקריאת ה-next ש"מביאה" לנו את האיבר מהגנרטור הראשון, עד שהאיבר אכן יעלה מהגנרטור השני ויהיה אפשר להחזיר אותו. הדבר תלוי מאוד בסידור של הגנרטורים.
לסיכום, בפונקציה זו ייתכן ויעבור מספר רב (שאינו סופי) של קריאות next על הפונקציה, מבלי שיוחזר שום איבר. הצורך לוודא את קיום האיבר בשני הגנרטורים עלול לקחת מספר קריאות רב מאוד על שני הגנרטורים, אפילו מבלי שנוכל להחזיר איבר אחד בפונקציית הגנרטור.
זאת בשונה מהפונקציה בסעיף א', שכבר בהרצה הראשונה תחזיר איבר, ולכל היותר תחזיר איבר נוסף בפעולת ה-next השלישית.
לכן לא ניתן ליצור פונקציה כזאת.

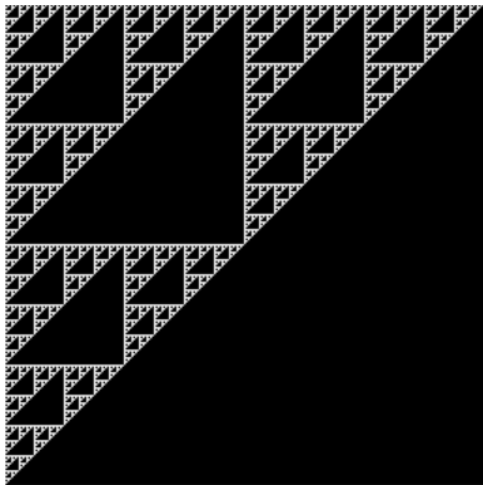
ג. ניתן ליצור פונקציה כזאת. נממש את הפונקציה בצורה הבאה. עבור כל מספר הגדול ממאה, נבדוק האם הוא ראשוני, לפי הסדר בסדרה חשבונית עולה (הפרש 1, אפשר גם הפרש 2 על מספרים אי-זוגיים). נבנה רשימה של מספרים ראשוניים עד 100, ואז נחלק את המספר הנבדק בכל אחד מהמספרים מהרשימה. אם השארית היא 0, נדע שהמספר הוא לא ראשוני ולא נחזיר אותו. אם עבור כל המספרים ברשימה השארית שונה מאפס, נוסיף את המספר לרשימת הראשוניים ונחזיר אותו. לאחר מכן נתקדם הלאה בסדרה החשבונית.

שאלה 6

א. להלן התמונה שהתקבלה:



ב. להלן התמונה שהתקבלה:



ג. להלן תמונת תעודת הזהות:

3 1 5 3 2 8 9 6 3