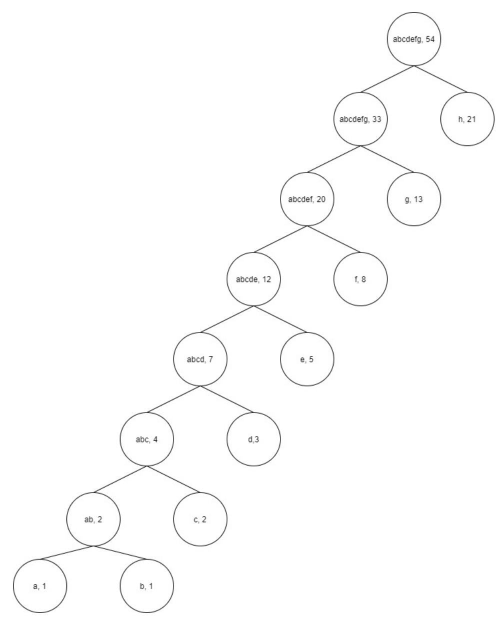
מבוא מורחב למדעי המחשב - תרגיל 6

רועי בוגין - 209729524, בר צדוק - 211964515

שאלה 2.



א.

נראה כי הקוד הנ"ל, הינו קוד האפמן תקין ואופטימלי עבור המקרה המתואר בשאלה.

a	1111111
b	1111110
c	111110
d	11110
e	1110
f	110
g	10
h	0

ב. נטען כי קוד ההאפמן האופטימלי כאשר התדירויות הן הn מספרי פיבונאצי הראשונים הוא עץ בו בן אחד הוא עלה עם מספר פיבונאצי הn, ומצד שמאל יש את העץ שיווצר עבור ה$n-1$ מספרי הפיבונאצי הראשונים, כאשר עבור $n=1$ העץ יהיה מספר הפיבונאצי הראשון (הגדרה בצורה רקורסיבית).

טענה זו נכונה כיוון שאם נסכום את הk מספרי פיבונאצי הראשונים, נקבל מספר שקטן ב1 ממספר פיבונאצי ה$k+2$, ולכן, בכל שלב הצמתים שיבחרו הן העץ שנוצר לפני (או מספר פיבונאצי הראשון) והמספר פיבונאצי ה$k+1$.

ג. נטען כי $|C(a_s)| - |C(a_t)| = 0$ לכל $s,t \in [n]$. נוכיח זאת באינדוקציה על $k > 0$ כאשר $n = 2^k$ ונתון ש$a_1 < a_2 < \cdots < a_n$ וכמו כן $a_n < 2a_1$. כאשר $k = 1$, נקודד אחד מהם ל0 ואת השני ל1. ולכן נקבל כי אכן $|C(a_1)| - |C(a_2)| = 1 - 1 = 0$ (אלה הם האופציות היחידות לבחירת s,t).

אם התנאי מתקיים עבור $k-1$, נוכיח כי התנאי מתקיים עבור k. נשים לב כי בשלב הראשון, נחבר את הקודקודים a_1, a_2, בשני a_3, a_4 וכן הלאה עד a_{n-1}, a_n, כיוון שעבור כל j מתקיים $a_{2j+1} + a_{2j+2} > 2a_1 > a_n$, ולכן הצמתים בעלי הערכים הקטנים ביותר יהיו a_{2j+3}, a_{2j+4}. נראה כי לאחר ביצוע כל שלבים אלו, נקבל 2^{k-1} קודקודים כאשר תדירות כל קודקוד j היא התדירות של $b_j = a_{2j-1} + a_{2j}$ ומתקיים $b_j = a_{2j-1} + a_{2j} > a_{2j-3} + a_{2j-2} = b_{j-1}$ וגם $b_{2^{k-1}} = a_{n-1} + a_n < 2a_n < 4a_1 < 2a_1 + 2a_2 = 2b_1$ ולכן קוד זה עונה על הנחת האינדוקציה, ומתקיים

$$|C(a_s)| - |C(a_t)| = \left(\left|C\left(b_{\lceil \frac{s}{2} \rceil}\right)\right| + 1\right) - \left(\left|C\left(b_{\lceil \frac{t}{2} \rceil}\right)\right| + 1\right) = \left|C\left(b_{\lceil \frac{s}{2} \rceil}\right)\right| - \left|C\left(b_{\lceil \frac{t}{2} \rceil}\right)\right| = 0$$

לכן, בפרט, עבור $s = 1, t = n$, הראנו כי ההפרש $|C(a_1)| - |C(a_n)| = 0$ לכל $n = 2^k$ ובפרט עבור $n = 256$.

ד. נטען כי $|C(a_1)| - |C(a_n)| = 1$.

אם נבצע את התהליך הרקורסיבי שתואר לעיל, בכל סבב מספר הקודקודים יתחלק ב2, בסבב ה2, כאשר נגיע למספר קודקודים השווה ל75, נסמנם $b_1, \ldots, b_{75}$ נשים לב כי a_1 נמצא בעץ b_1 בעומק 2 וa_{300} נמצא בעץb_{75} בעומק 2. נחבר תחילה את b_1, b_2 ולאחר מכן b_3, b_4 וכן הלאה עד b_{73}, b_{74} מאותן סיבות כמו בסעיף ג. כאשר נגיע לb_{75}, הקודקוד בעל התדירות הכי קרובה אליו יהיה הקודקוד (b_1, b_2), ולכן הוא יחובר אליו. ולכן העומק של הקודקוד b_{75} יהיה אחד פחות מהעומק של b_1, ולכן $|C(a_1)| = |C(b_2)| + 2 = |C(b_1)| + 1 + 2 = |C(a_n)| + 1$, ומכאן נובע כי

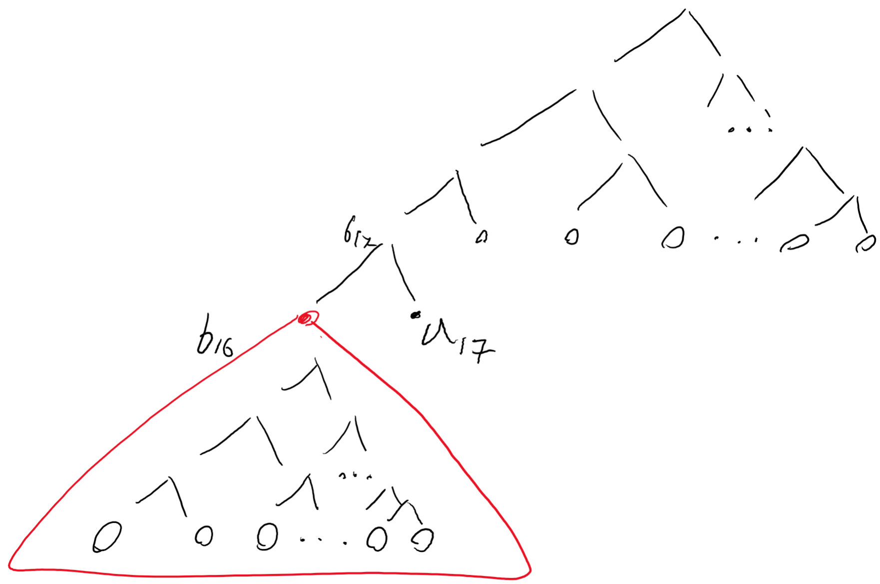
$$|C(a_1)| - |C(a_n)| = 1$$

ה. נטען כי $|C(a_1)| - |C(a_n)| = 5$.

נשים לב כי הצמתים $a_1, \ldots, a_{16}$ מקיימים את המקרה שהוכחנו בג'. ובנוסף, $\sum_{i=1}^{16} a_i < 16a_{16} < a_{17}$. לכן, תחילה יבנה העץ שמכיל את כל הצמתים $a_1, \ldots, a_{16}$. לאחר מכן, נסמן b_{16} להיות הצומת שמכילה את העץ של$(a_1, \ldots, a_{16})$. נשים לב כי $b_{16} < a_{17}$ ולכן צמתים אלה יתחברו לצומת שאותה נסמן b_{17}. כעת, ידוע כי $a_n < 2a_{17} < 2b_{17}$ ובנוסף נראה כי $a_n < 2a_{17} < 2a_{18}$ לכן אם נסדר את $b_{17}, a_{18}, \ldots, a_n$ על פי גדליהם, נקבל 256 צמתים אשר עונים על תנאי סעיף ג'. ולכן לכל צומת יהיה אותו מספר הביטים. כלומר, $|C(b_{17})| - |C(a_n)| = 0$ וכיוון שa_1 עלה שלb_{17} בעומק 5 (כיוון שהוא עלה של b_{16} בעומק 4 וb_{16} נמצא בb_{17} בעומק 1), נקבל כי

$$|C(a_1)| - |C(a_n)| = |C(b_{17})| + 5 - |C(a_n)| = 5$$

סקיצה:



שאלה 3.

א. נראה כי עבור הקלט $'aaaa'$, הפלט של שתי הפונקציות יהיה זהה, ושווה ל$['a', [1, 3]]$.

ב. טענה זו נכונה.

עבור הקלט $'aaabaaaab'$, נראה כי הפלט של שתי הפונקציות אינו זהה. בפונקציה החדשה יוחזר הפלט $['a','a','a','b','a',[5,4]]$ שיצוגו הבינארי באורך 58. ובפונקציה הישנה יוחזר הפלט $['a','a','a','b',[4,3],'a','b']$ שיצוגו הבינארי באורך 66.

ג. טענה זו אינה נכונה.

שתי הפונקציות לוקחות את התוצאה הטובה ביותר מבין הבדיקות שהן עושות. ברגע שהפונקציה הישנה מוצאת מקרה ששווה לכווץ, היא לוקחת אותו, ואינה בודקת אפשרויות חלופיות שיכולות להיות יעילות יותר בטווח הרחוק. לעומת זאת, הפונקציה החדשה בודקת את כל האפשרויות בהתאם לפרמטרים W, L, ולכן בפרט תבדוק כל אפשרות שהפונקציה הישנה בדקה ותחזיר תוצאה יעילה לפחות באותה המידה.

שאלה 4.

א.

(x_1, x_2, x_3)	$(x_1, x_2, x_3, x_1+x_2, x_1+x_3, x_2+x_3, x_1+x_2+x_3)$
$(0,0,0)$	$(0,0,0,0,0,0,0)$
$(0,0,1)$	$(0,0,1,0,1,1,1)$
$(0,1,1)$	$(0,1,1,1,1,0,0)$
$(1,1,1)$	$(1,1,1,0,0,0,1)$

ב. המרחק בין כל שתי מילים w_1, w_2 הוא 4.

נשים לב כי שינוי הביט הi של מילה משנה לי 4 ביטים בקוד תיקון השגיאות, וכי שינוי מ0 ל 1 או מ1 ל0 של הביט הi תמיד יהפוך את אותם הביטים.

לכן, אם על מנת להגיע ממילה w_1 לw_2 הפכנו k ביטים, ישתנו אותם הביטים כמו אם היינו מתחילים מהמילה $(0,0,0)$ ומשנים את אותם הביטים. על פי הטבלה שינוי של k ביטים מהמילה $(0,0,0)$ משנה 4 ביטים בקוד לתיקון שגיאות, ולכן גם אם נשנה k ביטים ממילה w_1, ישתנו בדיוק 4 ביטים בקוד לתיקון שגיאות.

ג. כן. ניקח $y = (0,0,0,0,1,1,0)$. הראנו כי כל מילה חוקית היא כולה אפסים, או במרחק 4 ממילת ה0. ולכן כיוון שבy יש 2 אחדות מרחקה של כל מילה מy הוא לפחות 2.

בנוסף הקוד של המילים $(0,0,0)$, $(0,0,1)$ שתיהן במרחק 4 מy ולכן y שבחרנו עונה על תנאי השאלה.

חלק שני משפט: bad_coding הוא קוד מטיפוס $[n = 4(|x|+1), k = |x|, d = 4]$ (נשים לב כי אם $|x| = 2$ המרחק המינימלי בין 2 מילות קוד תקינות הוא 8).

שאלה 5. נגדיר $C(i,j)$ להיות הסיבוכיות של הפונקציה fill_cell עבור התא הi,j. נראה כי

$$C(i,j) = \sum_{i+1}^{j} \sum_{l=1}^{|R|} \Theta(1) = \Theta((j-i+1)|R|)$$

כיוון שאנו עוברים על k אשר מקבל ערכים $i+1,\ldots,j$ ועבור כל k עוברים על כל ה$|R|$ חוקים כיוון ששליפה מסט מתבצעת ב$O(1)$ בממוצע. לאחר מכן ראינו כי שאר הקוד מתבצע ב$O(1)$ בממוצע ולכן גם ב$\Theta(1)$.

נחשב את הסיבוכיות של הפונקציה CYK (נסמנה $T(n)$)בהינתן הפונקציה $C(i,j)$ שחישבנו לעיל.

$$T(n) = \sum_{l=2}^{n} \sum_{i=0}^{n-l+1} C(i,i+l) = \sum_{l=2}^{n} \sum_{i=0}^{n-l+1} \Theta(l|R|) = \sum_{l=2}^{n} (n-l+1)\Theta(l|R|) = n\sum_{l=2}^{n} l\cdot\Theta(|R|) - \sum_{l=2}^{n} l^2\Theta(|R|) + \sum_{l=2}^{n} l\cdot\Theta(|R|)$$

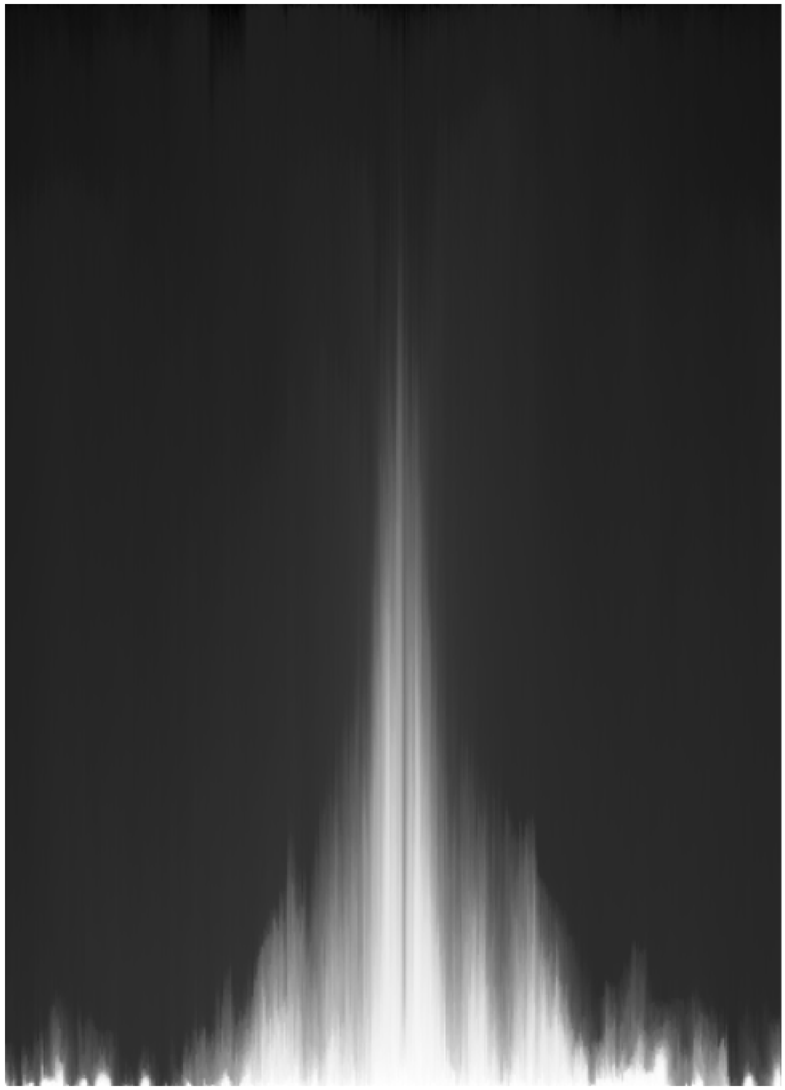
$$\Theta(|R|)\left(n\sum_{l=2}^{n} l - \sum_{l=2}^{n} l^2 + \sum_{l=2}^{n} l\right) = \Theta(|R|)\left(n\frac{n^2+n-2}{2} - \frac{n(n+1)(2n+1)-6}{6} + \frac{n^2+n-2}{2}\right) = \Theta\left(n^3|R|\right)$$

ולכן, $T(n) = \Theta\left(n^3|R|\right)$

שאלה 6.

א. הפונקציה what ממיינת כל עמודה בתמונה לפי בהירות. להלן מימוש שלנו לפונקציה והפלט שהתקבל

```
def what(img):
    img_cpy = img.copy()
    mat_cpy = img_cpy.load()
    w, h = img.size
    for x in range(w):
        s = sorted([mat_cpy[x,y] for y in range(h)])
        for y in range(h):
            mat_cpy[x,y] = s[y]
    return img_cpy
```



ב. התמונה המקורית:

פלט הפונקציה process_img ללא שינוי :

פלט הפונקציה process_img עם השינוי :

ההסבר לשינוי בפלט הוא שהפונקציה המקורית (ביחד עם פונקציית הlambda הנתונה) יוצרת עותק של התמונה ומשקפת אותה ביחס לציר ה-x ולכן מוחזרת תמונה משוקפת. בפונקציה החדשה, פעולת השיקוף מתבצעת על התמונה עצמה ולא על עותק שלה. לכן, החצי העליון של התמונה משוקף כמו בפונקציה המקורית אך החצי השני משוקף ביחס לתמונה שכבר שוקפה ולכן החצי התחתון של התמונה נראה כמו החצי התחתון של התמונה המקורית.