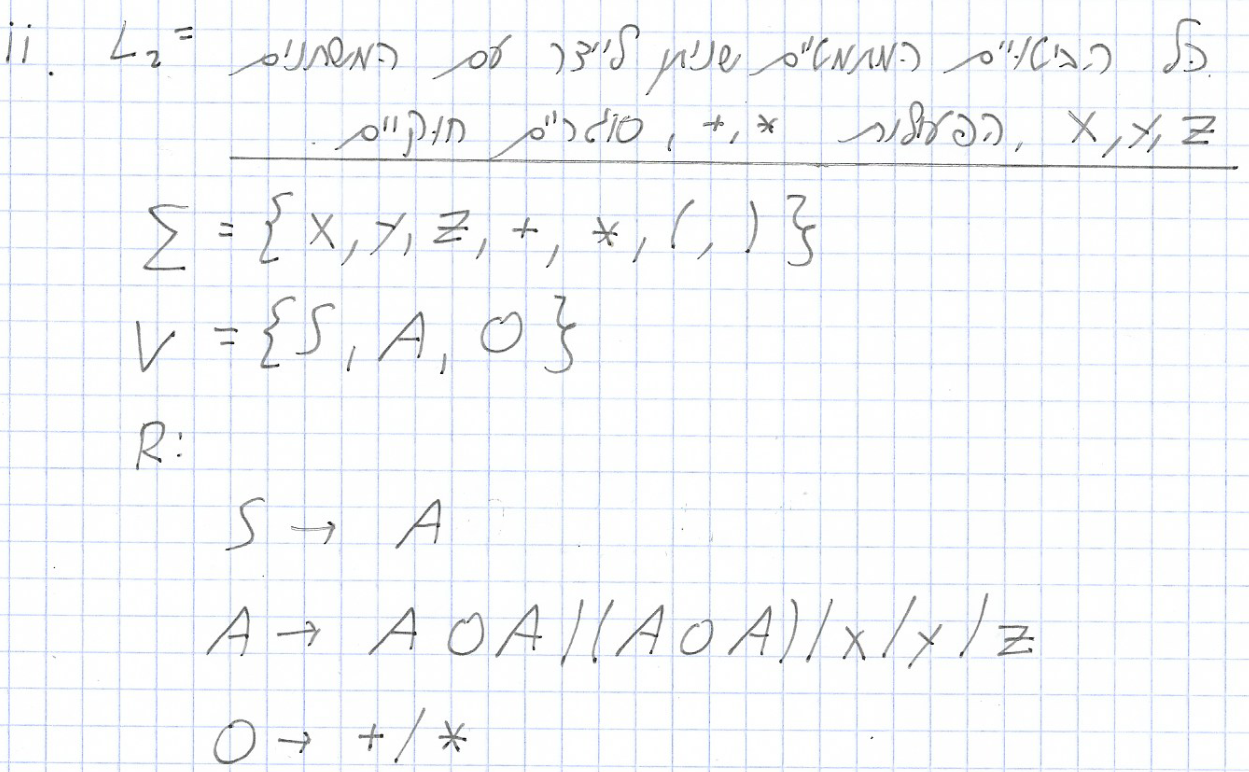
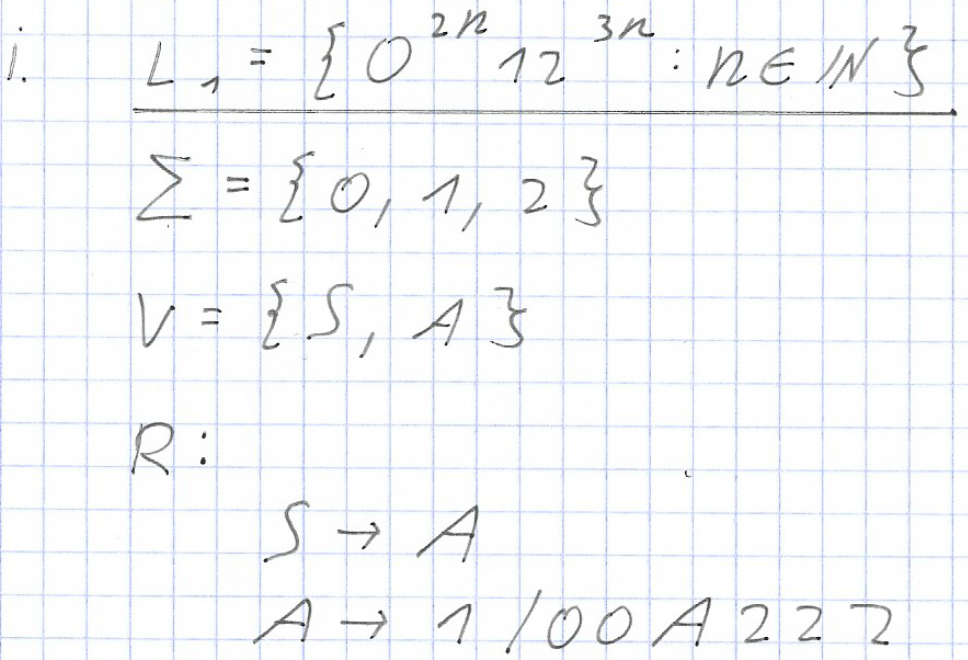
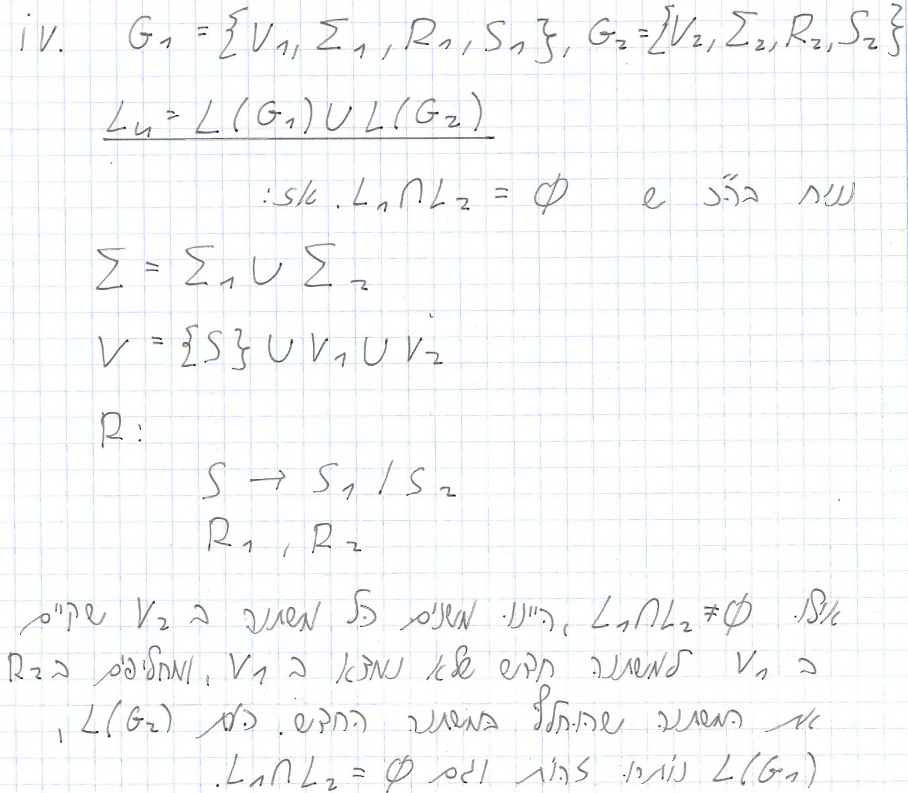
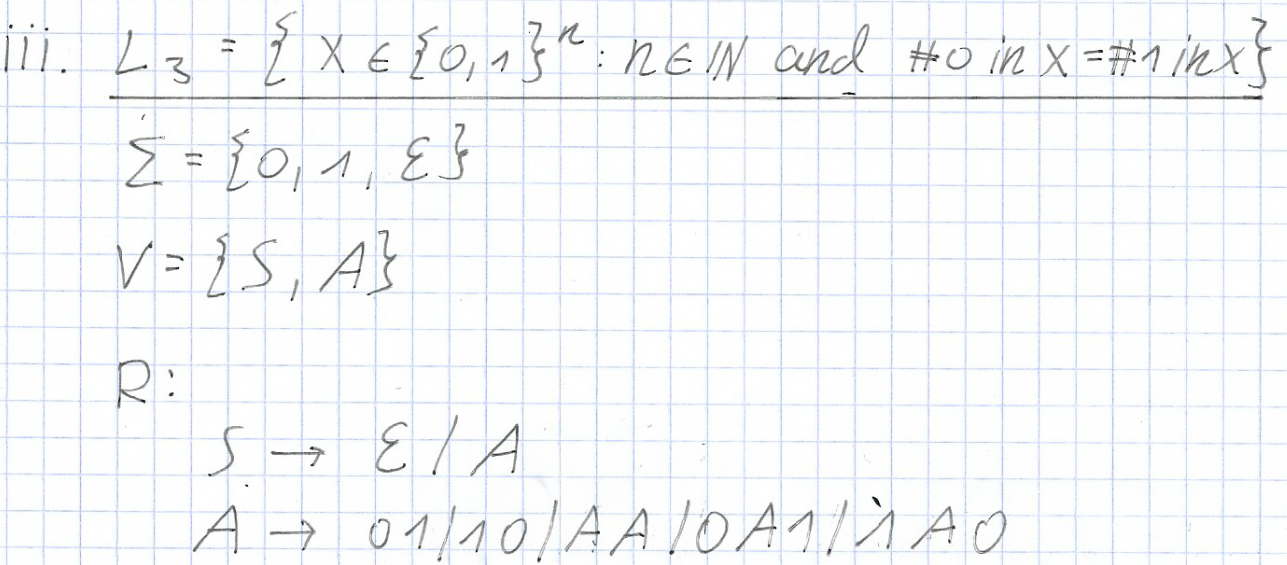
**שאלה 1**

סעיף א'





* את v לא הצלחתי.

הערה: בסעיפים ב' וג' התקשתי מאוד לתת הוכחה פורמלית בלי להסתבך, לכן ניסיתי להסביר את החלק המרכזי של הרציונל.

סעיף ב'

1. תשובה 3: **לכל הפחות אקספוננציאלי בk**. במקרה הגרוע שבו כל משתנה ניתן לגזור לצמד משתנים או לתו, נקבל נוסחת נסיגה:

f(k) = f(1)\*f(k-1) + f(2)\*f(k-2) + … + f(n-1)\*f(1)

שהיא איננה פולינומיאלית ב k כיוון שאילו הייתה עבור k מקבוע כלשהו, עבור k+1 נקבל משתנה נוסף לנוסחת הנסיגה בסתירה לכך שקיים חסם עליון פולינומיאלי (חישוב פשוט של נוסחת נסיגה ליניארית הומוגנית עם r משתנים, הסתירה היא למעלת הפולינום של החסם העליון). מכאן לא נותר אלא להסיק שהסיבוכיות היא לפחות אקספוננציאלית בk.

1. תשובה 2: **לכל הפחות אקספוננציאלי בk**. אמנם הממואיזציה חוסכת רבות, אבל בסופו של דבר כדי ליצור מבנה נתונים בגודל לפחות אקספוננציאלי בk, דרוש מספר אקספוננציאלי של פעולות.

סעיף ג'

הפונקציה מחזירה את מספר המחרוזות באורך k שיש בשפה של הדקדוק שהתקבל.

1. תשובה 3: **לכל הפחות אקספוננציאלי בk**. מבנה הפונקציה זהה לפונקציה שבונה את השפה בסעיף ב, ההבדל נמצא בנק שבה בסעיף ב' מוסיפים את המחרוזת לשפה. כאן פשוט מגדילים מונה ב1. אם כן מאותם שיקולים כמו בסעיף ב, נבצע את הפעולה של ההעלאה ב1 מספר פעמים לפחות אקספוננציאלי בk (זהה לסיבוכיות הזיכרון של ב' 1).
2. תשובה 2: **לכל היותר פולינומיאלי בk**. כאן בשונה מסעיף ב' הממואיזציה משמעותית גם בעניין הסיבוכיות עצמה. הממואיזציה גורמת לדילוג על ספירות שכבר בוצעו. בקצרה נאמר שהיא חותכת את החישוב של כל ה-

f(k-2) ומטה בנוסחת הנסיגה. מחישוב של נוסחת נסיגה נקבל ביטוי שאינו ליניארי, אך הוא גם איננו אקספוננציאלי כיוון שיש מספר סופי של מחוברים ומבנה הCNF הופך את החישוב שלהם לפשוט.

**שאלה 2**

סעיף ג'

לא ניתן לבנות כיוון שיתכן שהגנרטור שנקבל יהיה מורכב רק ממספרים שליליים ולכן לעולם לא נוכל להחזיר את המספר הבא שגדול מ0. במצב זה לגנרטור לא תהיה השהייה סופית כנדרש.

סעיף ה'

לא ניתן לבנות כיוון שיתכן שהגנרטורים שנקבל יהיו בכלל זרים (לעולם לא יחזירו את אותו הערך, לדוגמה g1 יוצר את סדרת המספרים השליליים -1, -2, -3, … וg2 יוצר את סדרת המספרים החיוביים 1, 2, 3, …). במצב זה לגנרטור לא תהיה השהייה סופית כנדרש.

**שאלה 3**

סעיף א'

1. מספר העלים יהיה t. כל תו מצוות לתו אחר או צומת, ומתחבר לשורש חדש של העץ. לאורך כל האלגוריתם שורש העץ הוא היחידי שמתעדכן, ולכן לא יתווספו בנים אל התו עד סוף התהליך. לבסוף נקבל שכל תו הוא עלה בעץ, וכל עלה בעץ הוא תו.
2. משקל השורש של העץ יהיה n. משקל של צומת בעץ הוא סכום משקלי הבנים, וכך גם נקבע משקל השורש. סה"כ, נקבל שמשקל השורש יהיה סכום משקלי העלים, שהם כפי שכבר ראינו בדיוק הא"ב של הקורפוס, ומשקל כל אחד מהם הוא מספר המופעים שלו בקורפוס. סה"כ יש n מופעים של תווים בקורפוס.
3. גובה העץ המקסימלי יהיה t. יתקבל במצב שבו אם נסדר את התווים לפי סדר התדירות שלהם נקבל שכל תו גדול מסכום התדירויות של כל התווים לפניו. במצב זה נחבר בכל שלב את הצומת היחידה עם עלה חדש משמאל (בהתחלה נחבר את שני העלים הקטנים ביותר). סה"כ כל צומת בעץ מחזיקה צומת מימין ועלה משמאל, בתחתית מוחזקים שני עלים, והשורש נספר גם הוא בחישוב הגובה. מכאן נקבל שהגובה הוא t.

גובה העץ המינימלי יהיה ערך שלם תחתון של log(t). יתקבל במצב שבו כל סכום תדירויות של שני תווים עולה על התדירות של כל תו אחר בקורפוס. במצב זה כל שני תווים יתחברו ביחד לכדי צומת, ומכאן והלאה נאלץ לחבר בכל פעם שני צמתים מאותו הגובה. סה"כ נקבל עץ בינאי (עד כדי עלה, תלוי בזוגיות מספר התווים בא"ב), שיחלוק את אותו מספר הקומות בעץ כמו העץ שהיה מתקבל אילו היינו מתעלמים מהתווים עם התדירות הכי נמוכה עד שהיינו מקבלים כמות תווים שהיא חזקה של 2 במספר שלם (מכאן מגיע הערך שלם תחתון).

1. מספר הצמתים הינו t-1. הוכחה באינדוקציה שלמה על t. הבסיס t=2 ברור, צעד האינדוקציה יכסה את כל התת מקרים שבהם מתקבלים לבסוף שני צמתים, כל מקרה נבדל מהאחר מהזוג הלא סדור של כמות העלים של שני הצמתים, ומוכח באמצעות האינדוקציה השלמה (נפרט קצת: נניח כמות העלים של צמת אחד היא k<=t, אז כמות העלים של הצמת השני היא t-k. מהנחת האינדוקציה לצמת הראשון יש k-1 צמתים ולצמת השני t-k-1, נאחד אותם לקבלת העץ הסופי, קיבלנו עוד צומת בעץ שהוא השורש, וסה"כ t-1 צמתים בעץ).

סעיף ג'

1. הפרכה: c = {‘b’: ‘1’, ‘c’: ‘10’, ‘a’: ‘110’}, text1=’a’, text2=’bc’.

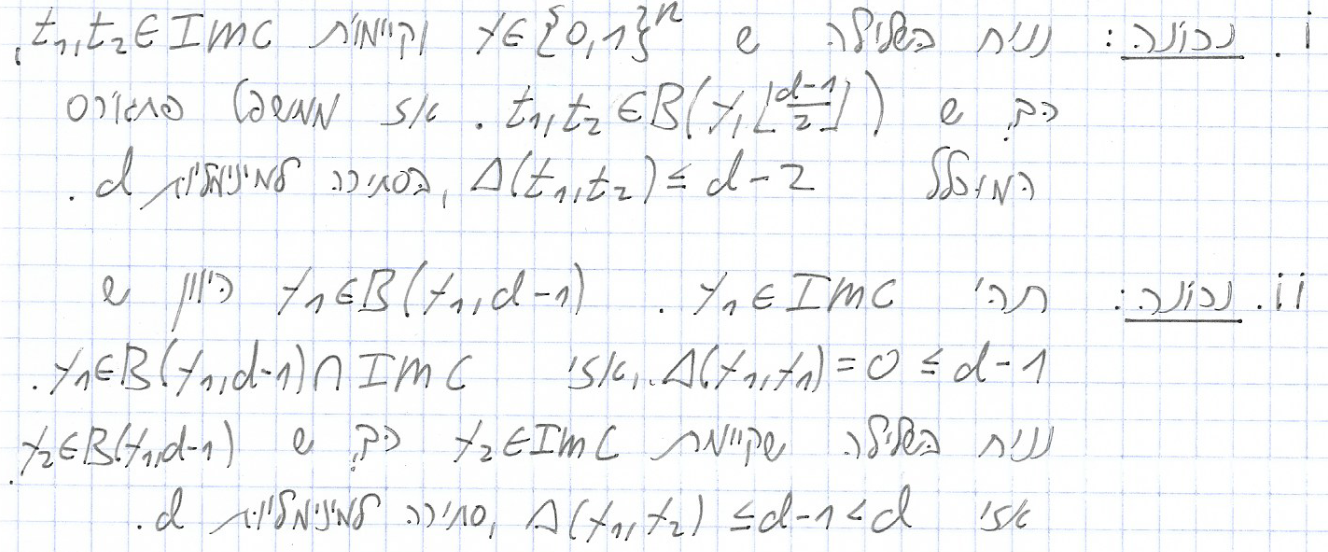
נקבל ש bin1 = ‘0110’, bin2 = ‘01010’.

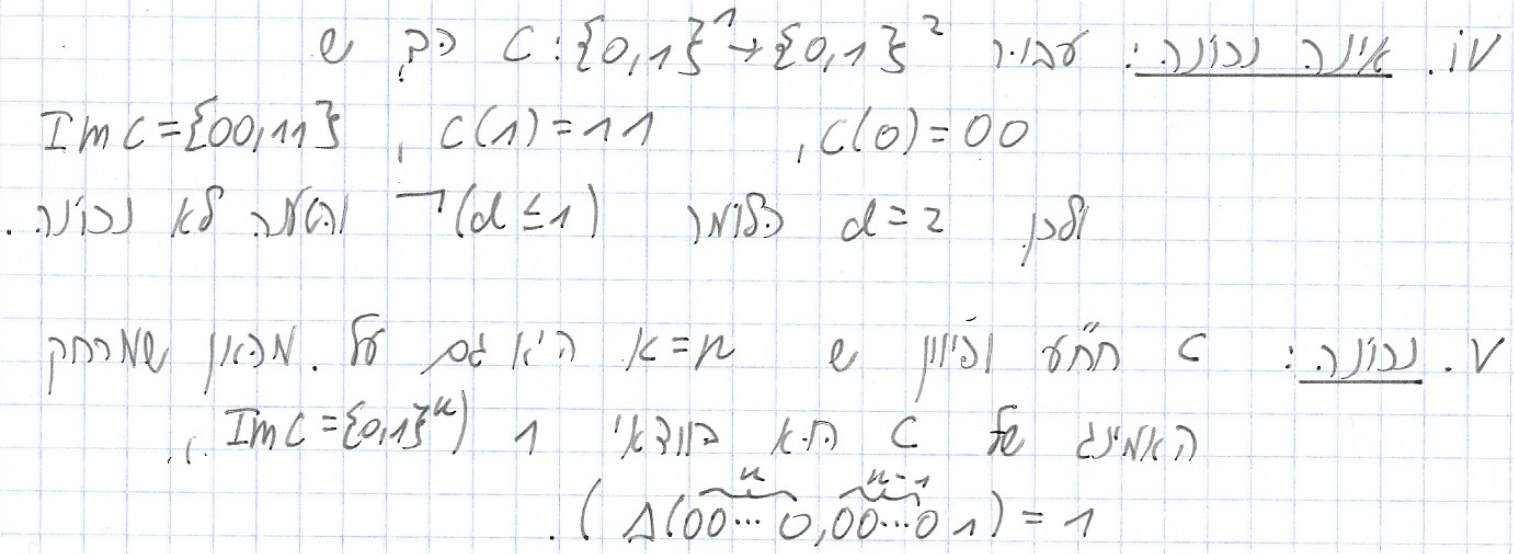
1. הפרכה: c = {‘b’: ‘10’, ‘c’: ‘110’, ‘a’: ‘111’}, text1=’ab’, text2=’cb’.

נקבל ש bin1 = ‘0111010’, bin2 = ‘0110010’.

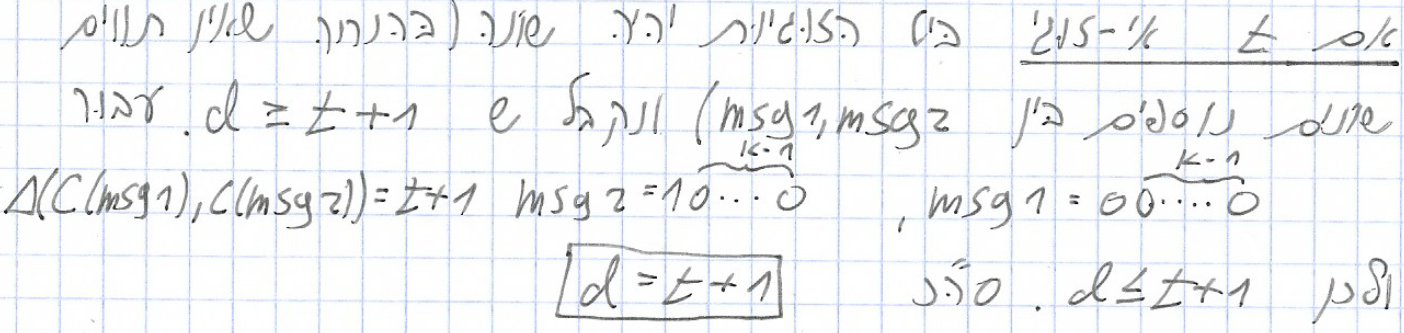
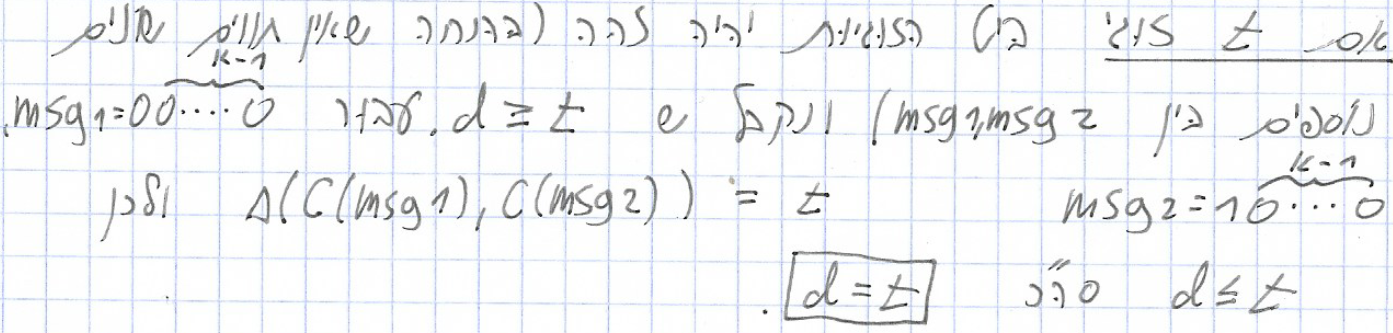
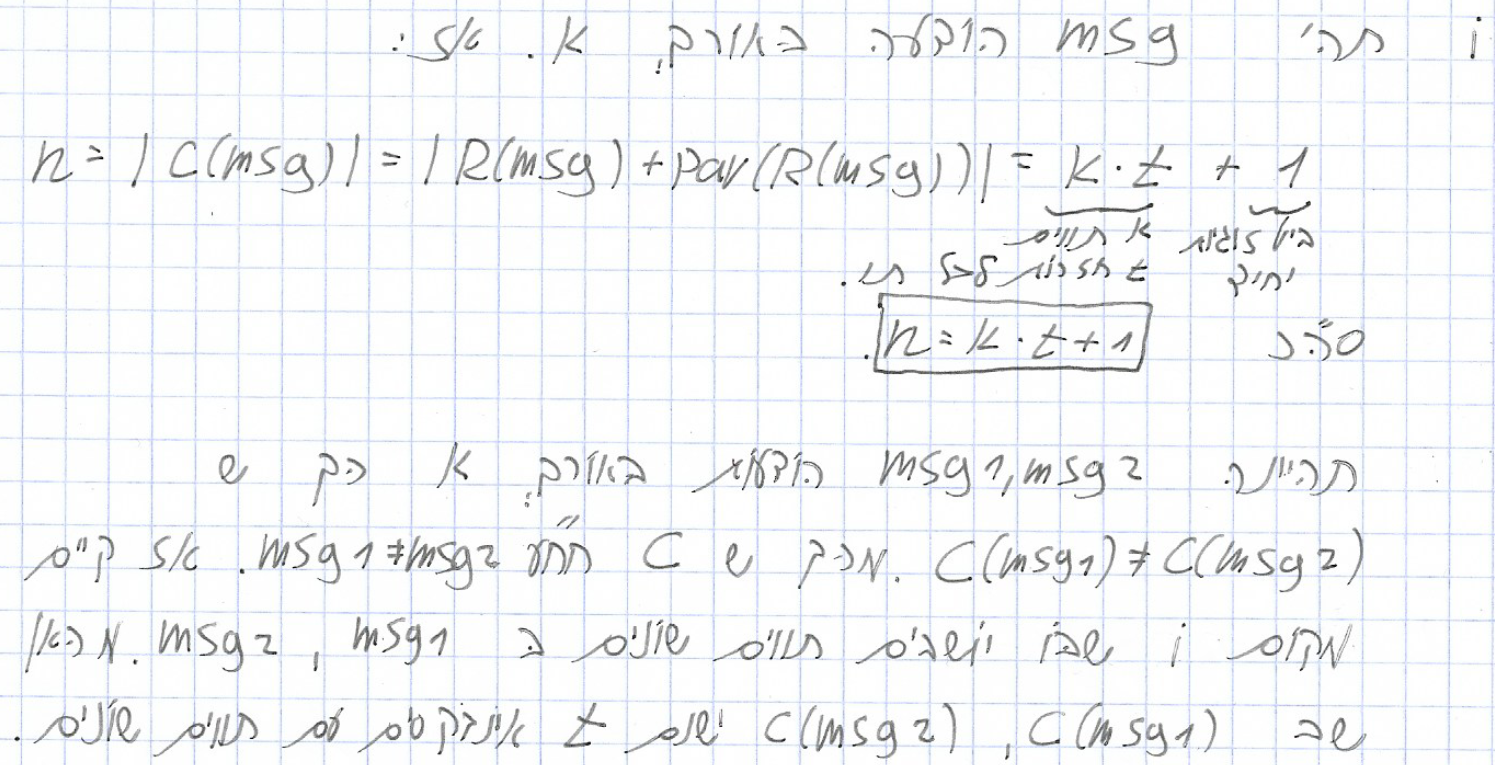
**שאלה 4**

סעיף א'





סעיף ב'



את סעיפים ii, iii, לא הצלחתי.

**שאלה 5**

סעיף ג'

|  |  |  |
| --- | --- | --- |
| 142 | 85 | 142 |
| 85 | 0 | 85 |
| 142 | 85 | 142 |

|  |  |  |
| --- | --- | --- |
| 255 | 255 | 255 |
| 255 | 0 | 255 |
| 255 | 255 | 255 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 0 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 |