**חלק 1 -סעיף א**  
  
על מנת לעבד קובץ לוג גדול בצורה יעילה, בחרתי לחלק אותו לטווחים (scopes) של בייטים במקום לפצל אותו לקבצים פיזיים נפרדים. כל תהליך מקבילי מקבל תחום קריאה מוגדר מראש — הוא מתחיל ממיקום מסוים בקובץ, קורא עד לסופו של התחום.   
גישה זו חוסכת בזמן ובמשאבי דיסק, מאחר שהיא נמנעת מיצירת קבצים זמניים מיותרים, ומבצעת פחות פעולות קלט/פלט (I/O). מספר התחומים נקבע לפי מספר הליבות הזמינות במחשב, כך שניתן לנצל בצורה מיטבית את יכולות העיבוד המקבילי של המערכת.  
  
**ה. סיבוכיות זמן ומקום**

נסמן:  
c- מספר הליבות הזמינות במחשב  
N- מספר השורות בקובץ  
k- מספר סוגי השגיאות  
n- הפרמטר הנתון  
  
   
סיבוכיות זמן  
הקוד עובר על כל הקובץ פעם אחת בלבד , אמנם בזכות ה multiprocessing יש מעבר במקביל על חלקים שונים של הקובץ אך הדבר לא משפיע משמעותית על סיבוכיות זמן הריצה שהיא פונקציה של גודל הקובץ.  
חילוק ל-scopes (זניח)  
מעבר על הקובץ- לכל התהליכים יחד  
מיזוג ה- counters מיזוג של c מילונים בגודל 0(k) - (לרב זניח)  
מציאת n השכיחים על ידי ערימה -O(klogn)  
ברור שN הוא הדומיננטי ביותר  
סה"כ

(c קבוע)  
  
סיבוכיות מקום  
כל תהליך קורא את השורות שלו **שורה-שורה**, לא טוען את כל הקובץ לזיכרון  
שימוש ב f.readline() שומר רק שורה אחת בזיכרון בכל רגע.  
כל תהליך שומר מונה שגודלו O(K)-רשימה של כל קודי השגיאה הייחודיים שהוא נתקל בהם.  
 עבור c תהליכים נקבל: O (ck), ולאחר המיזוג   
מציאתn השגיאות עם השכיחות הגבוהה על ידי שימוש בערימה בגודל n:   
החישוב לא כולל מקום קובץ הקלט כמובן שלא נחשב לחלק מסיבוכיות המקום!!  
סה"כ:  
  
   
   
 בהנחה ש n<=k