**תרגיל בית – הדסים 5**

**חלק א'**

**סעיף א'**

5. **ניתוח סיבוכיות:**

1. **פיצול הקובץ** - O(M) כאשר M הוא מספר השורות בקובץ.
2. **ספירת שגיאות בכל חלק** - O(K) כאשר K הוא מספר השורות בחלק.
3. **מיזוג הנתונים** - O(M) כי אנו עוברים על כל הקבצים ומאחדים.
4. **חישוב N הגדולים ביותר** - O(E log N), כאשר E הוא מספר קודי השגיאה הייחודיים.

**זיכרון:** הפתרון שומר חלקים קטנים בזיכרון ולא את כל הקובץ, ולכן הוא יעיל לשימוש בקבצים גדולים.

**סעיף ב**'

3. הרעיון של פתרון לשאלה הזאת הוא להקים מערכת שתוכל לעבד נתונים המגיעים בזמן אמת (streaming data) ולחשב את הממוצעים השעתיים בצורה דינמית, מבלי להמתין לכל הנתונים כדי לבצע את החישוב.

הנה איך אפשר לחשוב על פתרון:

1. **קבלת הנתונים בזמן אמת**: במקום לקרוא את הנתונים מקובץ סטטי, הנתונים יגיעו בצורה רציפה (streaming), כלומר כל רגע נתון יש מידע חדש שמגיע.
2. **אחסון וארגון הנתונים**: יהיה צורך לשמור את הנתונים הזורמים בצורה כזו שנוכל לעבוד איתם. אפשר להשתמש במבנה נתונים כמו queue או מאגר בזמן, שמאפשר לאחסן את הנתונים שמתקבלים.
3. **חישוב ממוצעים שעתיים בזמן אמת**: במקום לחשב את הממוצע רק לאחר סיום כל הקובץ, הפתרון יכלול מנגנון שיבצע את החישוב השעתי תוך כדי קבלת הנתונים. לכל נתון חדש שמגיע, נעדכן את הממוצע עבור השעה המתאימה.
4. **שמירה על חלון זמן**: כדי לחשב ממוצע שעתיים, נשתמש ב-"window" או "time window" שמכיל את כל הנתונים של השעה הנוכחית. כל פעם שמגיע נתון חדש, נוודא שהוא משויך לשעה הנכונה. אם עבר זמן יותר משעה מאז הנתון הראשון בחלון הזמן, נחשב את הממוצע, נשמור את התוצאה ונאפס את הנתונים עבור השעה הבאה.
5. **פרופורציה על זיכרון ומידול**: מאחר ונתונים זורמים יכולים להיות בלתי מוגבלים, יש להקפיד על שימוש בזיכרון באופן יעיל (למשל, על ידי שמירת רק הנתונים של השעה הנוכחית ושמירת הממוצעים הקודמים).
6. **עדכון ותגובה בזמן אמת**: ברגע שנחשב ממוצע, המערכת יכולה להוציא את התוצאה בזמן אמת (לשלוח את הממוצע החדש למערכת אחרת או לאחסן אותו במסד נתונים).
7. **חיזוי או עיבוד נוסף**: אם יש צורך, ניתן להוסיף שלב נוסף שמחשב מדדים נוספים על הנתונים, כמו חציון, סטיית תקן, או כל מדד אחר, ומעדכן אותם בזמן אמת.

הפתרון הזה דורש שמירה על מבני נתונים מותאמים לעבודה עם זרימות (streaming) ועל תכנון שמאפשר חישובים מתמשכים ללא עיכוב גדול בזמן, תוך שמירה על ביצועים גבוהים.

4. **התאמת הקוד לפורמט Parquet:**

1. **טעינת הנתונים** – במקום pd.read\_csv(), נשתמש ב- pd.read\_parquet().
2. **שמירת קבצים** – במקום to\_csv(), נשתמש ב- to\_parquet().
3. **נשמור על תאימות** – נוסיף אופציה לבחור באיזה פורמט לשמור את הקבצים.

**יתרונות של אחסון הנתונים בפורמט Parquet:**

✔ **גודל קובץ קטן יותר** – דחיסה טובה יותר מ- CSV.  
✔ **גישה מהירה יותר** – קריאה וכתיבה יעילים יותר.  
✔ **תמיכה בסוגי נתונים מורכבים** – שומר על טיפוסי נתונים בצורה טובה יותר.  
✔ **טעינה חלקית** – ניתן לקרוא רק עמודות מסוימות במקום לטעון את כל הקובץ.  
✔ **מתאים לעיבוד מקבילי** – משפר ביצועים במערכות Big Data.

**חלק ג'**

1. 2
2. a. מה​ו אופן השידור ‌בין השלט למ​זגן?

b.

c.