באחד השיעורי בית בjava ,נתנו לנו בעצם כתרגיל לקחת מערכת פשוטה שתבצע קריאת קבצים שונים לתוך מבנה נתונים משותף ששם נספרים מספרי המופעים של כל מילה ולבסוף להדפיס את מבני הנתונים הזה מהמיין. על המטלה היה להתבצע בעבודה מקבילית (Threads). בשיעורי בית הראשונים נתקלנו בבעיה בעצם , שכולם מנסים לכתוב לאותו מבנה נתונים במקביל ואז מה שקורה זה שהמידע לא נשמר כראוי .לדוגמא הראשון מנסה להגדיל את השדה הראשון בטבלה. אך בגלל שהמנגנון של העבודה המקבילית לא נשלט בידינו מה שיכול לקרות זה בעצם שהוא יקרא את הנתון יעשה חיבור של פלוס אחד אבל עוד לא יצרוב אותו לזיכרון . ואז יקפוץ למקום אחר שגם הוא במקרה משנה את המקום הזה , ואז יחזור לתהליך הקודם ויעדכן אותו בערך שאינו מעודכן יותר . תחילה בשיעורי הבית הראשונים כשידענו רק חלק מהחומר פתרנו את הבעיה ע"י כך שאמרנו לכל תהליך שיחכה שהשני יסיים את עבודתו ואז לא תהיה התנגשות . אך מה שבעצם קרה זה שאיבדנו את אפקט המקבילות .כשנדרשנו לשדרג את הפרויקט כך שיתמוך במקבילות , לקחנו את קטע הקוד שמעדכן את מבני הנתונים המשותף וגרמנו לו להיות מסונכרן משמע קריאה וכתיבה לטבלה תתבצע בבת אחת וכך בעצם כן שמרנו על מקביליות והכול היה אמור להתרחש כמו שצריך. אך כשהרצנו בכל פעם גילינו שהוא מדפיס טבלה ריקה . הצגנו למרצה והוא אמר שאכן השימוש במנגנון היה נכון והוא לא רואה את הטעות . כשניסיתי לעשות עיכוב זמן גדול לפני ההדפסה גילית שאכן אני רואה את הטבלה ! ואז הבנתי שאכן השתמשנו נכון אך המידע לא נשמר בטבלה או שהמידע עוד לא מופיע בטבלה משום מה . מה שגיליתי זה שגם למיין עצמו יש תהליך עצמאי , ובעצם הוא לא מחכה לסיום הזנת הנתונים לטבלה אלה ישר הולך להדפיס את הטבלה! ולכן היא ריקה . מה שעשיתי זה הוספתי לולאה שתמתין לסיום הזנת הנתונים ובכך פתרתי את הבעיה.