## Question 4

1. העקרון שמופר פה הוא עקרון ה Dependency inversion כיוון שEmployee כותב ישירות לAXL\DataBase ולא משתמש בinterface. שימוש בממשק מתאים יאפשר להוסיף בעתיד בקלות רכיבים נוספים שיכולים לכתוב לXML או ל DaraBae. בנוסף, עם תכנון נכון, אולי יהיה גם קל יותר להוסיף בעתיד אופציה של כתיבה למקומות אחרים (למשל אם שני הממשקים הללו יקבלו את המידע שכם צריכים לכתוב כמחזורת, הוספה של ממשק כתיבה נוסף שמקבל מחרוזת תהיה כמעט מיידית)



1. עקרון האחריות היחידה קשור ל Chain of Responsibility כי במקום לכתוב תוכנית על שעושה את כל המשימה הגדולה, שימוש הגיוני של שימוש בשרשרת אחריות הוא פיצול המערכת לרכיבים אשר לכל רכיב אחריות מסויימת אחת שהוא עושה ואז הוא מעביר את המשך המשימה הלאה לרכיב הבא, ובכך לכל רכיב רק אחריות אחת ויחידה.
2. דוגמא לעקרון היוצר בתכן שלנו הוא שהאינסטנס של Dish שמשתמש בExtra הוא זה שיוצר אותו. זה מתאים לדרישות כיוון שהוא עונה על אחד או יותר מהתנאים (על 3 מהם):

* Dish מכיל מופעים של Extra.
* מופעים של Dish משתמשים באופן הדוק בExtra.
* למופעים של Dish יש את המידע המשמש לאתמול Extra.

דוגמא לשימוש בעקרון בעל המידע היא שחישוב מחיר המנה מתבצע ע"י Dish ולא על ידי מישהו אחר (מלצר למשל). Dish מכילה את המידע על המחיר הבסיסי שלה ואיזה תוספות יש לה ומחשבת את המחיר הכולל שהוא מחיר המנה + מחיר כל התוספות שלה.

