עבודת הגשה 3:

חלק א': שאלות מילוליות:

1. א. דוגמא ליחס הורשה: מחלקה GregorianDate ו JewishDate יורשות ממחלקת MyDate_(שהיא מחלקה אותה יצרנו המאגדת מספר מחלקות שאופן מימושם זהה גם במחלקת GregorianDate וגם במחלקת JewishDate).
ב. דוגמא ליחס הפשטה: מחלקת MyDate שיצרנו, היינה מחלקה אבסטרקטית.
ג. דוגמא ליחס הכלה: מחלקת DateTime מכילה את מחלקות IDate ו Time כשדות.
ד. דוגמא לדריסה: במחלקות בהן השתמשנו במתודת ()toString, בוצעה דריסה. (הדריסה למתודה של מחלקת object).
ה. דוגמא לפולימורפיזם: במתודה after אשר מקבלת טיפוס IDate ובמחלקות GregorianDate ו JewishDate שממשות את IDate בצורה שונה.
ו. דוגמא להעמסה: הבנאים של מחלקות הexceptions, שמם של הבנאים זהה אבל כול בנאי מקבל פרמטרים שונים.

2. כן אפשרי להגדיר יחסי הורשה בין שתי המחלקות, משום שכל אחת מממשת את IDate כך שאם JewishDate תירש מ GregorianDate אזי יהיה ניתן להרחיב את המחלקה ולהוסיף לה מתודות ושדות נוספים בהתאם לצורך.

3. לא, מכיוון שנוצרים שני אובייקטים שהכתובות שלהם שונות, ולא הוגדר comparator אזי ההשוואה תעשה ע"י ה comparator הדיפולטיבי שהוא של מחלקת Object שהוא השוואה ע"י כתובות.
ולכן נקבל 2 איברים בset.
על מנת שקטע הקוד ידפיס 1, אפשר לבצע השמה ליצירת האובייקט כדוגמא:
GregorianDate g = new GregorianDate(10,10,2020);
ולהעביר את המשתנה g בצורה הבאה, ל ;setDates.add(g).

4. א. ע"י מתודה מחזירה את האובייקט עצמו (כך שמוחזרת הכתובת של האובייקט) ולכן יהיה ניתן לגשת אליו ולשנותו.
ב. מחלקות התאריכים ניתנות לשינוי משום שיש להם setters ואילו מחלקות ה exceptions לא ניתנות לשינוי משום שאין להם את מתודות ה setters.
ג. היתרון של immutable הוא הגנה על האובייקט משינוי ומניעת חריגות, החיסרון הוא שנצטרך ליצור מחלקות ייעודיות נוספות אם נרצה לבצע שינוי (getters, setters).
ד. כן, משום שמתודות הsetters מונעות השמה של תאריך שאינו נכון ובכך מונעות הדפסת תאריך שגוי שמתודת ה()toString עלולה להדפיס.