תוכנה 1 – תרגיל 10

חלק ג'

ב-SuperClass:

- הסרתי את final מהצהרת הclass בכדי שאפשר יהיה לרשת ממנו. כיוון שב-main של SubClass משימים אובייקט מטיפוס SubClass לתור SuperClass צריך להתקיים ירושה בכיוון זה. השגיאה הייתה בהצהרה על SubClass, בהצהרה extends SuperClass.
- הסרתי את הפרמטר Point p הניתן לבנאי עמ"נ שאפשר יהיה לקרוא לבנאי דיפולטי מה-main של SubClass – כאשר קוראים לבנאי הדיפולטי של SubClass אז דיפולטית נקרא ממנו הבנאי הדיפולטי של SuperClass. השגיאה הייתה ב-SubClass בפונקציית main בקריאה ()new SubClass.

ב-SubClass:

- שיניתי את הנראות של foo מ-private ל-protected כיוון שאי אפשר להפחית נראות של פונקציה דורסת בירושה. השגיאה הייתה בהצהרה על הפונקציה foo ב-SubClass.
- הסרתי את זריקת Exception מהמתודה foo כיוון שאי אפשר להכליל זריקת Exception גבוה יותר בהיררכיה ממה שהוצהר בSuperClass, וכן שהמימוש בSubClass אכן לא זורק שום Exception. השגיאה הייתה בהצהרה על הפונקציה foo ב-SubClass, בהצהרה throws Exception.
- הוספתי זריקת IOException מ-main (כולל import ל-Exception זה) כיוון שכאשר קוראים ל-foo של האובייקט c, בזמן קומפילציה מתייחסים לאובייקט זה לפי הגדרתנו כאובייקט מטיפוס SuperClass, והמימוש של foo בSuperClass עשוי לזרוק IOException, ולכן הייתה שגיאה בקריאה של c.foo(-20) של Exception לא מטופל (למרות שבפועל הפונקציה שתיקרא היא foo של SubClass אשר לא זורקת Exception.
- הוספתי .super בתוך הפונקציה foo בהשמה של Point p1 = p, כיוון שהשדה p שנמצא בSubClass הוא מטיפוס int ולא מטיפוס Point, ואילו השדה p שנמצא ב-SuperClass (כלומר super.p) הוא מטיפוס Point. השגיאה הייתה בשורת ההשמה המדוברת.
- שיניתי את ה-casting בתוך הפונקציה foo בהשמה של p1.x = (double) b מ-double ל-float כיוון שהשדה x של Point הוא מטיפוס float ולא ניתן להשים double לתוך float. כמו כן, כיוון ש-b הוא int, בוודאות ניתן להמיר אותו ל-float ואין סכנה לשגיאה בזמן ריצה. השגיאה הייתה בשורת ההשמה המדוברת.

ב-Point:

- הסרתי את abstract מהצהרת הclass בכדי שאפשר יהיה לייצר אובייקט מסוג Point ב-SuperClass. השגיאה הייתה ב-SuperClass, בשורה המצהירה על השדה p ומאתחלת אותו ב-()new Point.

לאחר התיקון:

התכנית מייצרת אובייקט מסוג SubClass ומשימה אותו לתוך אובייקט מסוג SuperClass. אז, היא קוראת בזמן ריצה לפונקציית foo שלו (של SubClass) עם פרמטר 20-, ומדפיסה את תוצאתה. פונקציית foo המדוברת מחזירה תמיד 0.0 אם ניתן לה מספר חיובי, או 0.0- אם ניתן לה מספר שלילי (כיוון שתמיד ה-Point המאותחלת בה, p1, נוצרת עם ערך דיפולטי שלא משנים, p1.y = 0, ולכן תוצאת המכפלה שתוחזר תמיד תיתן סוג של 0). במקרה שלנו ניתן לה הערך 20- ולכן היא מחזירה 0.0-, וזהו הערך שמודפס למסך.

חלק ד'

- נתתי שמות אינדיקטיביים לפונקציות ולמשתנים לאורך התכנית – חשוב כיוון שכעת מאוד ברור רק משמות הפונקציות והמשתנים מה התפקיד של כל אחד מהם, והרבה יותר קל לקרוא את התהליך.
- הפרדתי את תהליך קבלת הקלט מהמשתמש לפונקציה – חשוב כיוון שכעת התהליך קורה במקום אחד עבור 2 קבלות הקלט ואין שכפול, וכן כי אם נרצה להחליף את הקלט במשהו אחר נוכל לעשות זאת בקלות ברגע שהקלט מופרד מהלוגיקה.
- הפרדתי את פונקציות כתיבת הפלט לפונקציות נפרדות כלליות אשר מסוגלות לטפל בכל סוג set/map – חשוב ברמת התכנית הנוכחית מאותה סיבה שכתבתי קודם, וכן כי בצורת ההפרדה הזו נוכל לעשות שימוש עתידי בפונקציות אלו בתכניות עתידיות כיוון שהן כלליות – רק נצטרך לשנות את הגדרת הפרטיות ואולי להוציא אותן לclass של Utils.
- הפרדתי את 2 סוגי הניתוח השונים שנעשים ל-2 פונקציות טיפול – חשוב כיוון שבעצם יש לנו כאן 2 תכניות חבויות באחת, וצורת ההפרדה הזו משקפת זאת באופן נכון יותר, ומקלה על מעקב והתמודדות עם בעיות.
- הוצאתי את הטיפול בשגיאות ואיחדתי אותו במקום אחד ב-main, כך שבתוך הפונקציות כל פונקציה דואגת למשאבים שלה בלבד – חשוב כיוון שקודם לכן הטיפול היה לא אחיד, מה שגורר בעיות תחזוקה, קריאות, ומגדיל את הסכנה לטעות.