# HW1

**טופז אהרון 305302127**

**יקטרינה פודיאצ'ב 316741149**

## שאלה 1

### סעיף ב

1. במקרה זה, אם מתקיים: gs.p2 == this.end יש להפוך את הסגמנט ולבצע את המימוש המקורי עליו(בעזרת המתודה `reverse` של GeoSegment).

2. המפרט החדש חזק יותר מהמפרק המקורי מכיוון שהדרישות של פסקת ה-requiers@ מקלות יותר(פסקה חזקה יותר, פחות דרישות) effects@ זהה עבור קלטים שעומדים ב-requiers@ המקורי.

### סעיף ג

במקרה זה, אם מתקיים: gs.p2 == this.end יש להפוך את הסגמנט ולבצע את המימוש המקורי עליו(בעזרת המתודה `reverse` של GeoSegment).

המפרט החדש חזק יותר מהמפרק המקורי מכיוון שהדרישות של פסקת ה-requiers@ מקלות יותר(פסקה חזקה יותר, פחות דרישות) ואפקט זהה עבור קלטים שעומדים ב-requiers@ המקורי.

### סעיף ד

GeoSegment אינו true subtype של Rout מכיוון שהספציפיקציה של `addSegment` ב-GeoSegment היא חלשה יותר מב-Rout .

Rout אינו true subtype של Rout מכיוון ש-Rout אינו מכיל מתודת `getName`

### סעיף ה

נגדיר מחלקה חדשה IsraelTouristFormatter שתהווה צאצא של המחלקה, RouteFormatter. המחלקה תממש את המתודה האבסטרקטית של המחלקה computeLine בצורה כזו שתחזיר מחרוזת שתייצג משפט בשפה העברית שמכוון מה heading המקורי אל ה- geoFeature המבוקש. המפרט של המטודה הנ"ל ישאר זהה למפרט של המתודה במחלקת האב RouteFormatter, ולכן תוכל להוות java subtype שלה ולכן הדוגמה תתאים להיררכיה הנ"ל.

נגדיר מחלקה חדשה OldWalkingRouteFormatter שתתן כיוונים בצורה דומה למחלקה RouteFormatter, אך תדרוש שאורך כל סגמנט בדרך לא יעלה על חצי קילומטר. המחלקה הנ"ל תממש את המטודה computeLine בצורה דומה למחלקה RouteFormatter, אך מאחר והמפרט שלה חלש יותר מהמפרט של המחלקה הנ"ל, לא תוכל להוות צאצא שלה, ולכן לא תירש ממנה אלא תהווה מחלקת אב בפני עצמה.

## שאלה 2

א) ברטוב

ב) הבעיה העקרונית של מימוש מסוג זה היא שאין אפשרות להכללה של RouteFormatterGUI.
המחלקה בנויה אך ורק עבור משתנה מסוג Route ו-subtypes שלו. בתוכנות גדולות נשאף לכתוב את הקוד כך שיתאים באופן רחב לכמה שיותר סוגי משתנים כדי לחסוך שכפול קוד.

ג) הפרמטר owner – מאחר שה GeoSegmentsDialog הוא תת חלון של ה GUI הראשי, אנו נדרשים לספק לבנאי שלו את ה owner, ועליו אנחנו מציגים את תיבת הדיאלוג הנוכחית. ה frame הוא ה container שמכיל את כל המאפיינים הגרפיים של החלון המקורי ושל ה Dialog שבו אנו בוחרים את הסגמנטים להוספה. כל GUI חדש שאנחנו יוצרים מתווסף ל container הזה ולכן הוא נחוץ.

הפרמטר pnlParent – זו למעשה מחלקת ה RouteFormatterGUI שאל שדותיה אנחנו מעוניינים להוסיף סגמנטים חדשים, ובדיוק מסיבה זו, מאחר שה GeoSegmentDialog מספק GeoSegments חדשים לחלון GUI הראשי, ואנו מעוניינים להעביר את הסגמנט שבחרנו בדיאלוג אל מטודת ה addSegment של ה RouteFormatterGUI, האירוע של לחיצת כפתור ה add צריך לקבל את האובייקט של חלון האב כדי לקרוא למטודה זו ולכן הוא פרמטר הכרחי לבנאי.

ד) כדי לתמוך בתכונה זאת עלינו לדרוש את השינויים הבאים:

מפרט והמגדיר את פעולת המחיקה של סגמנט מתוך route, (במחלקה route).
הוספת פונקציה removeSegment ל-RouteFormatterGUI אשר תקרא לפעולת מחיקה של סגמנט מה-rout במידה והוא קיים.
הוספת כפתור מחיקה של סגמנת ב-GeoSegmentsDialog שבעת לחיצתו תקרא הפונקציה removeSegment שתוארה לעיל.

## שאלה 3

### סעיף א

Abstraction function: The polynomial that this class represents is the sum of all the PolyTerms in the list terms where each PolyTerm represents a monomial with degree that eaquals to power and a coefficient that equals coeff .

### סעיף ב

Representation invariant: terms is a list of terms in which the last monomial power is always the largest in the list. Every monomial in the list has coeff != 0 unless it's power == 0.

Terms always contains at least one monomial.

### סעיף ג

Representation invariant: terms is an ordered list of terms in the sense that for every PolyTerm term1 and PolyTerm term2, if term1 comes before term2 in terms then term1.power > term2.power. Every monomial in the list has coeff != 0 unless it's power == 0. Terms always contains at least one monomial.

כלומר, לא נאפשר כמה מונומים עם דרגה זהה וגם נשמור על המערך ממוין לפי הדרגה. כך נוכל לארוך חיפוש בינארי ולהחזיר את המקדם ברגע שנמצא מונום עם דרגה מתאימה.