דרור ונטורה 301394631
אלדר דמארי 300014529

שאלה 4

1. גם בסינטקס "עשיר" (ML) וגם בסינטקס "פשוט" (scheme) ניתן לממש את אותם הדברים כאשר בסינטקס "עשיר ישנה נוחות גדולה יותר תוך שימוש בהרבה מילים שמורות וקיצורים Syntactic Sugar המוגדרים במערכת, ובסינטקס "פשוט" נצטרך להגדיר אותם בעזרת ביטויים פשוטים.

2. ב-scheme פרט לטיפוסים הפרימיטיבים המוגדרים במערכת אין תמיכה בטיפוסים נוספים ועל מנת ל"הגדיר" טיפוס נצטרך לבנות לו מעטפת חיצונית (ADT) מתויג כלומר Tagged-Data כאשר בML ישנה הגדרת טיפוסים מובנית לכל טיפוס בסיסי המוגדר בשפה ובנוסף להגדיר טיפוסי משתמש באמצעות הפקודה datatype.

3. ב-scheme ה-type checking מתבצע בזמן ריצה בלבד (עבור הטיפוסים המוגדרים בשפה) כלומר באופן דינמי.
כאשר ב ML בדיקת הטיפוס מתבצעת בזמן קומפילציה (באופן סטטי) ואילו ניתן לכפות על משתנה להיות מטיפוס מסויים למשל ע"י כתיבת תבנית מסויימת שתקבל משתנים רק מאותו טיפוס.

4. גם ב scheme וגם בml ניתן להעביר פרמטרים לפרוצדורות, כאשר גם הפרמטרים המועברים יכולים להיות פרוצדורות בעצמן. ההבדל הוא שבML ניתן לכפות על פונקציה לקבל "טיפוס" מסויים כפרמטר ובמידה הפרוצדורה לא קיבלה ערך מתאים תזרק שגיאה לפני הרצת הקוד (בזמן קומפילציה) ואילו ב scheme לא ניתן לעשות זאת ולכן רק בזמן ריצה/שערוך הפרוצדורה תזרק שגיאה במידה ואין התאמת טיפוסים.

5. ב-scheme הרשימות הן הטרוגניות, כלומר ניתן להכיל טיפוסים שונים בתוך אותה רשימה
כאשר ב ML הרשימות הן הומוגניות בלבד. הערה: ניתן ליצור רשימות גנריות בML אבל ברגע שטיפוס ספציפי מסויים נכנס לרשימה הרשימה הופכת להיות רשימה עבור אותו טיפוס בלבד.

6. ML תומכת בהגדרת טיפוסי משתמש, באמצעות פקודה מובנית datatype.
scheme אינה תומכת בהגדרת טיפוסי משתמש באופן רשמי, אבל על מנת ליצור טיפוסים כאלו ניתן להגדיר ADT עבור כל טיפוס משתמש חדש, כפי שצויין לעיל ע"י שימוש ב Tagged Data.

7. באופן דומה לשאלה 5, scheme תומכת ברשימות הטרוגניות בעוד ש-ML לא.