2.12.09

(1)

קומפילציה - תרגול 7

היום נדבר על בדיקת טיפוסים. למשל, השמת מחרוזת למשתנה int, השמה למשתנה לא מוגדר, פעולה מתמטית על משתנה בוליאני, קריאה לפונקציה עם משתנים לא מתאימים, וכו'.

<u>תזכורת:</u> בטבלת הסימנים מופיעים את כל המשתנים שמוגדרים בסקופ מסוים (וקישור לטבלה ברמה גבוהה יותר).

ל-exp אנחנו יכולים להגדיר טיפוס. אנו מציינים פורמלית את הטיפוס T של exp בכתיבה הבאה exp : T (למשל 2 : int, 2×3 : int).

להגדרה זו מוסיפים קונטקסט של סביבה. בכתיבה כזו, ניתן לדעת גם מה טיפוס של exp שמשלבים משתנים/פונקציות. סימון הסביבה מתבטא ע"י ⊢ [שם סביבה] (למשל E ⊢ exp : T).

אחד התפקידים שלנו הוא לעבור אחר exp ולוודא של כל אחד מהם מוגדר היטב לטיפוס מסוים (ומהו).

חוקי טיפוסים מוגדרים להיות [שם] $\frac{\text{תנאים}}{\text{מסקנה}}$. למשל, $\frac{E \vdash e_1 : int \quad E \vdash e_2 : int}{E \vdash e_1 + e_2 : int}$ [+]

מגדיר כי אם e_1 ו-e_2 הם מסוג int בסביבה E, אז חיבור בינהם בסביבה E נותן טיפוס מסוג int. ואלו קוראים [+].

אם כלל טיפוסים לא מוגדר תנאי, אזי הכלל מגדיר אקסיומה (למשל שליטרל 7 הוא מסוג int).

ל-statements אין שיעשהו ערך. הם מוגדרים היטב (מבחינת טיפוסים) בסביבה E מסוימת אך אינם מחזירים ערך. לכן נסמן E ⊢ S (ללא : סוג ערך).

הגדרת חוקים המשלבים statements מבוצעת בכתיבה דומה ל-exp.

מתודות יכולות להיות גם exp וגם statement (בתלות ההגדרה של הערך המוחזר ממנה type / void).

במידה והמתודה מחזירה ערך מסוים, יש לבדוק שהטיפוס שהיא מחזירה זהה לטיפוס שהיא אמורה להחזיר. אם היא אמורה להחזיר void, אזי אם מוחזר return ללא ערך, הפעולה חוקית.

נשים לב, שייתכן והמתודה אמורה להחזיר טיפוס T, והיא בפועל מחזירה T', הפעולה עשויה להיות חוקית אם T' יורש מ-T (נסמן T'≤T). עם זאת, נזכור כי גם אם A יורש מ-B, זה <u>לא</u> אומר ש $A[] \le B[]$!

Flow לניתוח סמנטי:

תחילה, יש להוסיף את מחלקת הספריה ל-AST של התוכנית. לאחר מכן, יש לבנות את טבלת הטיפוסים הגלובלית (טיפוסים מוגדרים מראש וטיפוסים מוגדרים ע"י משתמש). לאחר מכן, יש לבנות כניסות כל טבלאות הסימנים. בשלב הבא בודקים שכל משתנה מוגדר ומשתמשים איתו לעיקוב של בטבלת הסימנים. לסיום עוברים מלמטה למעלה ומוודאים חוקיות הגדרה היטב של טיפוסים.