מושגים בשפות תכנות, שנה"ל תשע"ה, סמסטר ב', מועד א'

פרופ' מולי שגיב, עודד פדון

הנחיות כלליות:

- משך הבחינה 3 שעות.
- מותר להשתמש בכל חומר כתוב (אין להשתמש במחשבון, מחשב, או פלאפון).
- בכל השאלות עליכם לתת הסבר משכנע מדוע התשובה שכתבתם נכונה, אלא אם מצויין אחרת.
- במבחן 7 שאלות. שאלה 1 היא חובה, ועליכם לבחור לענות על 5 שאלות מבין שאלות 2-7.
- את כל התשובות יש לכתוב במחברת הבחינה בלבד, תשובות על טופס הבחינה לא ייבדקו.
- חלוקת הניקוד: שאלה 1, 10 נקודות, שאלות 2-7, 18 נקודות כל אחת. סה"כ 100 נקודות.

חובה לענות על השאלה הבאה:

שאלה 1 - שאלת חובה (10 נקודות)

בשאלה זו עלייך לסמן נכון / לא נכון בכל אחד מהסעיפים (בשאלה זו אין צורך לנמק). תשובה נכונה מזכה ב 2 נקודות לכל סעיף, ותשובה לא נכונה לא משפיעה על הציון.

1. תחת static scoping, משתמשים ב access link כדי לגשת למשתנים לא לוקאליים.
2. אם תוכנית OCaml נדחית על ידי ה compiler בגלל טיפוסים לא חוקיים, אז בהכרח הרצת התוכנית תוביל לשגיאה דינאמית של התאמת טיפוסים.
3. JavaScript קיבלה את שמה בגלל הדימיון לשפת Java.
4. בשפת OCaml, בהנחה שהמשתנה x מכיל רשימה, הפקודה `let y = x` תגרום להעתקה של כל האיברים ברשימה לרשימה חדשה.
5. התוכניות הבאות שקולות תחת Structural Operational Semantics:

- `while true do skip`
- `abort`

בחרי 5 שאלות מהשאלות הבאות. משקל כל שאלה 18 נקודות.

שאלה 2 - Parsing

נניח שנרצה לתכנן שפת תכנות שתאפשר לבצע ניתוח תחבירי (parsing) בשיטת LL(1), ונניח שאנו רוצים לאפשר להגדיר משתנים בסגנון Java/C, לדוגמה המילים הבאות צריכות להיות חוקיות בשפה:

- `int x, y, z`
- `float x, y`
- `int a`

כמו כן, אנו רוצים שהניתוח התחבירי יפריד בין רשימת משתנים מטיפוס int ורשימת משתנים מטיפוס float. כלומר, הדקדוק צריך להכיל שני non-terminals:

IntIdList
FloatIdList

ששניהם יגזרו רשימות של שמות משתנים (מזהים) עם פסיקים ביניהם. שימי לב ש IntIdList ו FloatIdList גוזרים את אותן רשימות מזהים.

א. כתבי דקדוק חסר הקשר לשפה הנ"ל שיוצר את ההפרדה המתוארת בין רשימה של משתנים מטיפוס int ורשימה של משתנים מטיפוס float. הטרמינלים (tokens) בשפה הם:

id, ",", float, int

הדקדוק חייב להכיל לפחות את ה non-terminals הבאים:

S, IntIdList, FloatIdList

כאשר S הוא ה non-terminal התחילי (start symbol).

ב. האם הדקדוק שכתבת בסעיף א' הוא LL(1)? אם לא, הציעי דקדוק LL(1) שקול (עבור אותה שפה, ויוצר את אותה הפרדה בין IntIdList ל FloatIdList).

ג. הדגימי את הדקדוק שכתבת בסעיף ב' ע"י ציור עץ הגזירה למילה הבאה:

```
int x, y
```

ד. נניח שהיינו רוצים לכתוב דקדוק שייצור את ההפרדה בין IntIdList ו FloatIdList עבור שפה בסגנון Algol/Scala שמכילה הגדרות משתנים מהצורה:

```
var x, y, w : int
var z, t : float
```

הסבירי מדוע לא ניתן לבצע זאת בעזרת דקדוק LL(1)?

שאלה 3 - סמנטיקה

נרצה להוסיף לשפת While את הפקודה הבאה:

while b **odd do** S_1 **even do** S_2

זוהי לולאה שרצה כל עוד התנאי b מתקיים, באיטרציות אי-זוגיות מבצעת את S_1, ובאיטרציות זוגיות מבצעת את S_2 (האיטרציה הראשונה היא אי-זוגית). לדוגמה, התוכנית הבאה:

while n ≤ 10 **odd do** n := n+2 **even do** n := n+8

תסתיים במצב בו n=12 בהנחה שהיא מתחילה ממצב בו n=0.

א. הרחיבי את ה Natural Operational Semantics כדי לטפל בפקודה החדשה. הכללים אינם יכולים להסתמך על מבנה לולאת while הרגילה בשפה (כלומר לא ניתן להתייחס בכללים למבנה לולאת while הרגילה, אלא רק לפקודת הלולאה החדשה).

ב. הדגימי את הסמנטיקה המורחבת שהגדרת בסעיף א' ע"י בניית עץ גזירה לתוכנית מפסקת הפתיחה ממצב התחלתי בו n=0.

ג. הרחיבי את ה Structural Operational Semantics כדי לטפל בפקודה החדשה. הכללים אינם יכולים להסתמך על מבנה לולאת while הרגילה בשפה (כלומר לא ניתן להתייחס בכללים למבנה לולאת while הרגילה, אלא רק לפקודת הלולאה החדשה).

ד. הדגימי את הסמנטיקה המורחבת של סעיף ג' ע"י בניית סדרת גזירה לתוכנית מפסקת הפתיחה ממצב התחלתי בו n=0.

ה. האם ניתן לכתוב ביטוי בעזרת לולאת while רגילה ששקול סמנטית ללולאה החדשה? אם כן, כיצד? ואם לא, מדוע? (אין צורך להוכיח פורמלית, מספיק להסביר אינטואיטיבית).

שאלה 4 - λ -calculus

נתון הביטוי הבא ב λ -calculus:

$(\lambda s.\ \lambda z.\ z)\ ((\lambda s.\ \lambda z.\ s\ z)\ \lambda x.x)$

א. ציירי את ה AST המתאים לביטוי.

ב. כתבי סדרת חישוב (reduction) לביטוי תחת call by value semantics.

ג. כתבי סדרת חישוב (reduction) לביטוי תחת lazy evaluation semantics.

ד. הסבירי בקצרה את ההבדל בין call by value ו lazy evaluation.

שאלה 5 - OCaml

הבהרה: בשאלה זו אסור להשתמש בפונקציות ספרייה מהמודול List

א. כתבי פונקציית OCaml בשם total שמפעילה פונקציה על איברי רשימה ומחזירה את סכום התוצאות. הנה מספר דוגמאות לשימוש בפונקציה:

```
# total (fun x -> x) [1; 2; 3; 4; 5];;
- : int = 15
# total String.length ["Python"; "OCaml"; "JavaScript"; "Scala"];;
- : int = 26
# total int_of_string ["1"; "2"; "30"];;
- : int = 33
```

ב. הסיקי את הטיפוס הכללי ביותר של הפונקציה שכתבת בסעיף א'.

ג. מהי צריכת הזיכרון של הפונקציה שכתבת בסעיף א' כתלות באורך הרשימה שהיא מקבלת? הסבירי היכן זיכרון זה מוקצה, והאם ניתן להפעיל את הפונקציה על רשימה של מאה מיליון איברים?

ד. אם צריכת הזיכרון של הפונקציה שכתבת בסעיף א' אינה קבועה, כתבי גרסה שלה שיכולה לפעול בזיכרון קבוע ללא תלות באורך הרשימה (רמז: רקורסיית זנב). מותר להגדיר פונקציית עזר.

שאלה 6 - JavaScript

אחת הדרכים ליצור אובייקט עם שדות פרטיים ב JavaScript היא ע"י שימוש ב closures. הקוד הבא מדגים זאת, וממש אובייקט Point שיש לו מתודות move, toString, ואובייקט ColoredPoint שיש לו מתודות move, darken, toString:

```
function Point(x, y) {
   var obj = {};
   obj.move = function(dx, dy) { x += dx; y += dy; };
   obj.toString = function() {
       return "[Point with x=" + x + " and y=" + y + "]";
   };
   return obj;
}
function ColoredPoint(x, y, color) {
   var obj = Point(x, y);
   obj.darken = function(tint) { color += tint; };
   obj.toString = function() {
       return "[ColoredPoint with x=" + x + ", y=" + y +
              ", and color=" + color + "]";
   };
   return obj;
}
p = Point(3, 4);
p.move(1, 2);
console.log(p.toString());
```

א. הסבירי בקצרה את הרעיון של מימוש שדות פרטיים ע"י closures.

ב. בקוד הנ"ל יש באג. כתבי 3 שורות של JavaScript שמדגימות את הבאג, והסבירי מהו הבאג וממה הוא נובע.

ג. האם ניתן לתקן את הבאג ע"י שינוי הפונקציה toString ב ColoredPoint בלבד?

ד. תקני את הבאג, ע"י תוספות ל Point ושינוי הפונקציה toString ב ColoredPoint.

שאלה 7 - Types

נתון המימוש הבא של פונקציית OCaml שאמורה לבדוק האם רשימה אחת היא תחילית (prefix) של רשימה אחרת:

```
let rec is_prefix xs ys = match (xs, ys) with
 | ([], _) -> true
 | (_, []) -> false
 | (x::xs', y::ys') -> is_prefix xs' ys'
```

א. נתחי את הטיפוס הכללי ביותר של הפונקציה is_prefix.

ב. כיצד ניתן להבין מהטיפוס שיש באג במימוש?

ג. תקני את הבאג.

ד. נתחי את הטיפוס של הפונקציה המתוקנת.

בהצלחה,

מולי ועודד