מושגים בשפות תכנות, שנה"ל תשע"ה, סמסטר ב', בחינה לדוגמה

הנחיות כלליות:

- 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 dynamic scoping, משתמשים ב access link כדי לגשת משתנים גלובליים.
2. בכל שפות התכנות הכלליות הסטנדרטיות ניתן להגדיר את אותה קבוצה של פונקציות מעל המספרים הטבעיים.
3. ברוב שפות התכנות שתומכות ב exceptions, ה exception handler נקבע ע"י static scoping.
4. בשפות שמאפשרות להחזיר פונקציה אחת כערך חזרה של פונקציה אחרת, בהכרח אין אפשרות לשחרר את רשומת ההפעלה (activation record) כאשר הפונקציה מסיימת.
5. תוכניות בשפת Scala רצות מהר יותר מתוכניות בשפת OCaml בגלל השימוש ב JVM.

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

שאלה 2 - Parsing

נתבונן בדקדוק הבא:

```
S → ( L ) | a
L → L, S | S
```

כאשר S,L הם non-terminals, וה terminals הם: a, "(", ")", "," (כלומר a, פתח סוגריים, סגור סוגריים, ופסיק). S הוא המשתנה התחילי

א. האם הדקדוק הוא LL(1)? אם לא, הציעי דקדוק LL(1) שקול (עבור אותה שפה).

ב. בני את טבלאות ה FIRST, FOLLOW, SELECT של הדקדוק מסעיף א'.

ג. הדגימי את תהליך בניית עץ הגזירה ע"י recursive descent parser למילה הבאה:

```
(a,a)
```

שאלה 3 - סמנטיקה

נרצה להוסיף לשפת While את הפקודה הבאה:

```
repeat S until b
```

זוהי לולאה שתמיד מתבצעת פעם אחת לפחות, והביצוע שלה נפסק כאשר התנאי b מתקיים. לדוגמה, הקוד הבא:

```
repeat x := x-10 until x < 10
```

יסתיים במצב בו x=5 אם יתחיל במצב בו x=55, ויסתיים במצב בו x=-3 אם יתחיל במצב בו x=7.

א. הרחיבי את ה Natural Operational Semantics כדי לטפל בפקודת repeat. הכללים אינם יכולים להסתמך על מבנה לולאת while בשפה (כלומר לא ניתן להתייחס בכללים למבנה לולאת while, רק ללולאת repeat).

ב. הדגימי את הסמנטיקה המורחבת של סעיף א' ע"י בניית עץ גזירה לתוכנית הבאה:

```
y := 1; repeat y := 2*y until y > 5
```

ג. הרחיבי את ה Structural Operational Semantics כדי לטפל בפקודת repeat. הכלל(ים) אינם יכולים להסתמך על מבנה לולאת while בשפה (כלומר לא ניתן להתייחס בכללים למבנה לולאת while, רק ללולאת repeat).

ד. הדגימי את הסמנטיקה המורחבת של סעיף ג' ע"י בניית סדרת גזירה לתוכנית מסעיף ב'.

ה. האם ניתן לכתוב ביטוי בעזרת לולאת while ששקול סמנטית ללולאת repeat? אם כן, כיצד? ואם לא, מדוע?

שאלה 4 - lambda calculus

נתון הביטוי הבא ב lambda calculus:

```
(λx. λx. (λx.x) x) ((λx. x x) λx.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. כתבי סדרת חישוב (reduction) לביטוי תחת normal order semantics.

שאלה 5 - OCaml

הבהרה: בשאלה זו אסור להשתמש בפונקציות ספרייה מהמודול List

א. כתבי פונקציית OCaml בשם find שמחפשת איבר ברשימה ומחזירה את האינדקס של המופע הראשון שלו. אם האיבר לא נמצא הפונקציה זורקת (`Failure "Not Found"`). הנה מספר דוגמאות לשימוש בפונקציה:

```
# find 1 [4; 2; 1; 7; 1; 8];;
- : int = 2
# find 0 [4; 2; 1; 7; 1; 8];;
Exception: (Failure "Not Found").
# find "OCaml" ["Python"; "OCaml"; "JavaScript"; "Scala"];;
- : int = 1
# find "Haskell" ["Python"; "OCaml"; "JavaScript"; "Scala"];;
Exception: (Failure "Not Found").
```

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. מותר להגדיר פונקציית עזר.

שאלה 6 - JavaScript

נתון קוד ה JavaScript הבא:

```
1:    var a = 10;
2:    var h = function(x) { return this.a + a; };
3:    var obj = {a: 3, foo: h};
4:    var f = function() {
5:        var a = 30;
6:        var g = h;
7:        return (g(1) + obj.foo(2) + a);
8:    };
9:   alert("f() = "  + f());
```

א. מה תהיה התוצאה של הקריאה g(1) בשורה 7?

ב. מה תהיה התוצאה של הקריאה obj.foo(2) בשורה 7?

ג. מה תהיה תוצאת הקריאה ל f() בשורה 9?

ד. הסבירי את תשובותייך בסעיפים הקודמים באמצעות טבלה מפורטת שעוקבת אחר רצף הקריאות, ומסבירה בכל קריאה למה מתייחס כל משתנה.

שאלה 7 - Types

נתון המימוש הבא של פונקציית OCaml שאמורה לבדוק האם שתי רשימות שוות:

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

א. נתחי את הטיפוס של הפונקציה equal.

ב. כיצד ניתן להבין מהטיפוס שיש באג במימוש? הסבירי מה הפונקציה בעצם עושה.

ג. תקני את הבאג כך שהפונקציה equal תאפשר לבדוק האם שתי רשימות זהות, כלומר שוות איבר איבר.

ד. נתחי את הטיפוס של הפונקציה המתוקנת.

בהצלחה,

מולי ועודד