חלק 1 - תיאורטי

שאלה 1

לא. בשפה L3 אין חוק חישוב ייעודי עבור ביטויים מסוג let. בכל פעם שניתקל בביטויי מסוג זה, נצטרך לכתוב אותו מחדש כאפליקציה של ביטויים מסוג lambda. כלומר, ב-L3 let מהווה סוכר תחבירי בלבד, ללא חוק חישוב ייעודי.

שאלה 2

כן. עוד בשלב הפרסור אנחנו ממירים את כל צורות ה-let בקוד לצורת הפעלת למבדה שקולה. ובזמן חישוב של הפעלת למבדה, נוצר closure בעת חישוב האופרטור בביטוי. הן במודל הסביבות והן במודל ההצבה, יוצרים closure לאופרטור בביטוי ההפעלה השקול.

שאלה 3

דוגמה ראשונה - חילוק ב0

```
(/ 5 0)
```

דוגמה שנייה – הפעלת אופרטור פרימיטיבי על טיפוסים לא תואמים

```
(+ 5 "Hi")
```

דוגמה שלישית – פנייה למשתנים שטרם הוגדרו בתכנית

```
(define x 5)
(+ x y)
```

זו התכנית בשלמותה. כלומר y לא הוגדר ולכן תעלה שגיאה.

דוגמה רביעית – הפעלת פרוצדורה עם מספר פרמטרים לא תואם

```
((lambda(x) x) 5 4)
```

שאלה 4

הפונקציה valueToLitExp ממירה רשימת הערכים להצבה לביטויים ליטרליים המייצגים את ערכם. המרה זו נדרשת משיקולי תאימות טיפוסים שכן גוף הפרוצדורה להצבה מוגדר במושגים של CExp של הפרסר, בעוד שהערכים להצבה הם כבר במושגי ה-Value של האינטרפרטר.

שאלה 5

בחישוב בסדר נורמלי אין צורך ב-valueToLitExp מכיוון שכל הביטויים אינם מחושבים לפני ההצבה, כשמגיעים לשלב ההצבה, כל הביטויים הם עדיין מסוג CExp, ובמצב כזה לא נוצרת בעיה של אי תאימות טיפוסים.

שאלה 6

אופרטור פרימיטיבי הוא פרוצדורה המובנית בשפה, ומחושב על פי חוק החישוב הדיפולטיבי של הפעלת אופרטור על אופרנדים. למשל "+" , הארגומנטים עליהם הוא פועל צריכים להיות מחושבים לפני שמפעילים את אופרטור.

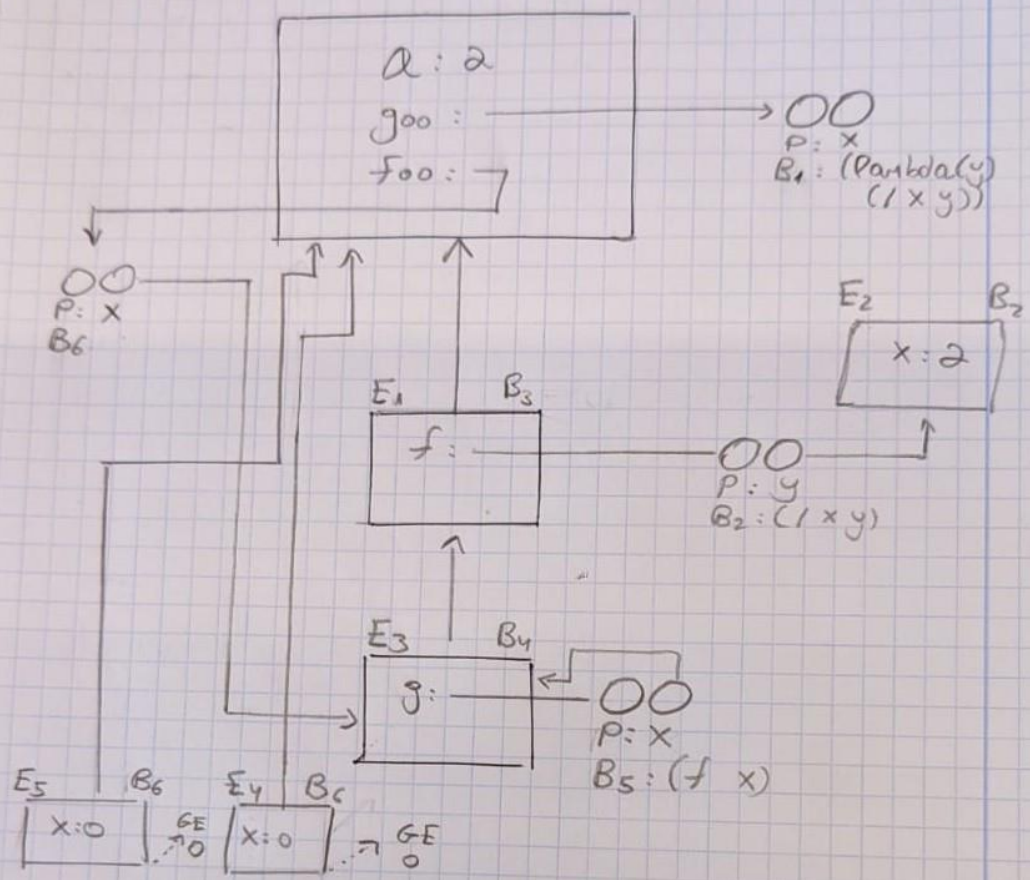
בעוד שצורה מיוחדת היא מבנה בשפה אשר דורש חוק חישוב מיוחד. למשל הצורה המיוחדת if, שדורשת חישוב של אחד מהביטויים then/else.

ההבדלים בין השניים נובעים מהגדרתם. יש להם הרבה מן המשותף, כמבנים בשפת תכנות. אך בסופו של יום, הם נבדלים בעיקר בחוק החישוב שלפיו הם מחושבים.

שאלה 7

במודל ההצבה, כל הפעלה של פרוצדורה (גם אם היא הופעלה כבר בעבר), כרוכה בשכתוב גוף הקוד שלה. שכתוב זה כולל פעולות נוספות כמו שינוי שמות כל הפרמטרים של כל הפרוצדורות המוגדרות בגוף, והחזרת ערכי הארגומנטים למבנה של ביטויים בגישה האפליקטיבית. במודל הסביבות, איננו נדרשים בפעולות מסוג זה. בקוד עם הרבה פונקציות עם גוף גדול, שימוש במודל הסביבות יחסוך המון עבודה.

1.8



שכתבה ל- let בשביל סימון מה של הגופים:

```
(define foo
  (let ((f (goo a))
        (let ((g (lambda(x) (f x))))
          (lambda (x)
            (if (= x 0)
                x
                (g x))))))
```

B5

B3

B4

B6