1. טיפוס הוא למעשה סט של ערכים שמאופיין על ידי קבוצה מובחנת של ביטויים. לדוגמא, טיפוס Boolean הוא טיפוס שיש לו 2 ערכים: true, false וניתן להפעיל עליו פעולות כמו : and,or,not. ערך ניתן להעביר לפרוצדורה וניתן לקבל מפרוצדורה, כאשר לכל ערך יש טיפוס מתאים.
2. מקטע a : במקטע a פונקציית define עושה binding בין x ל-5 בglobal environment **ולכן יש side-effect.**

מקטע b : אין side-effect

מקטע c: אין side-effect. מאחר ואיקס שונה מ-6, מורצת השורה (x x \*) ולא השורה של display.

1. Boolean
2. Procedure שהטיפוס שלה הוא: [Number \* Number → Number]
3. Number
4. Number
5. Void
6. Boolean
7. שגיאה – חלוקה באפס.
8. 2
9. שגיאה – x לא מוגדר ב-GE.
10. Void (נוצר משתנה חדש ב-GE)
11. <procedure>
12. 510
13. 42
14. 3
15. (define x 0)

(x /)

הsyntax נכון, אולם כאשר נריץ את השורות האלו נקבל שגיאה כי לא ניתן לחלק באפס.

1. (define a (lambda (x) (display x)))

((a 2) 3\*)

הsyntax נכון, אולם כאשר נריץ את השורות האלו נקבל שגיאה כי לא ניתן לכפול מספר בטיפוס מסוג void (הטיפוס של a הוא void על פי ההגדרה בשורה הראשונה).

* האופרטור cond ראשית בודק את הביטוי odd? 72 מכוון שהערך שלו הוא f# הוא לא מחזיר את f#. האופרטור עובר לבדוק את הביטוי הבא: f#, שוב מכיוון שהערך שלו הוא f# הוא לא מחזיר את ההדפסה whoops. כעת האופרטור יחזיר את ערך הביטוי האחרון שמוגדר אחרי ה-else. הביטוי מורכב מהאופרטור הלוגי or . כפי שלמדנו, ביטוי or מחשב את ערכי האופרנדים משמאל לימין ומחזיר את ערך הביטוי הראשון שאינו f# והיתר אינם מחושבים. לכן יוחזר המספר 7 מבלי שהמשך הביטוי יחושב (ולכן לא תוחזר הודעת שגיאה על חלוקה באפס). **התוצאה של כל הביטוי אם כן היא 7.**
* ביטוי and מחשב את ערכי האופרנדים משמאל לימין כל עוד ערכם אינו f#. אם מצא f#, הוא עוצר את החישובים ומחזיר אותו, אחרת מסיים את החישוב ומחזיר את ערכו של הביטוי האחרון. לכן, ראשית and בודק את הביטוי הראשון שערכו t# ולכן עובר לביטוי השני. הביטוי השני מורכב מהאופרטור הלוגי or אשר מחשב את הערך של (even? 3) שהינו f# ולכן עובר לחשב את הביטוי השני (lambda () (+ 1 2 3)) שערכו t# ולכן or יחזיר t#. האופרטור And עובר לחשב את הביטוי האחרון אשר ערכו 6. מכוון שכל הביטויים שand חישב אינם f#, האופרטור and יחזיר את הערך האחרון, 6. **התוצאה של כל הביטוי אם כן היא 6.**

1. אם or לא היה special form הוא היה מחושב כמו ביטוי מורכב סטנדרטי, כלומר, הוא היה מחשב את כל תתי הביטויים תחתיו באופן רקורסיבי ואז היה מופעל ערך ה-or על האופרנדים. חישוב זה היה גורר הודעת שגיאה כי הביטוי (1 0 /) אינו מוגדר (לא ניתן לחלק באפס).