שאלה 1

1. אין תוכנית ב־L1 שאי אפשר להמיר לשפהL11. define רק נותן שם לביטוי, ואפשר תמיד להחליף את השם בביטוי עצמו.
2. קיימות תוכניות ב־L2 שלא ניתנות להמרה ל־L21, למשל פונקציות רקורסיביות, כי define נחוץ כדי לאפשר לקרוא לפונקציה עצמה.
לדוגמה בL2 ניתן לכתוב פונקציה שמחשבת עצרת לכל מספר טבעי ובשפה L21 לא ניתן. כן ניתן לכתוב פונקציה שמחשבת עצרת למספר ספציפי אך לא פונקציה כללית שתעבוד לכל מספר.

```
(define fact
  (lambda n
    (if (= n 0)
        1
        (* n (fact (- n 1))))))
```

3. אין תוכנית ב־L2 שלא ניתנת להמרה ל־L22. L2 בכל מקרה מוגבלת לביטוי אחד וכל פונקציה מרובת פרמטרים ניתן לפרק לשרשרת של פונקציות חד־פרמטריות עם גוף של ביטוי אחד.
לדוגמה פונקציה שמקבלת שני פרמטרים תהפוך לפונקציה שמקבלת את הפרמטר הראשון ומחזירה פונקציה שמקבלת את הפרמטר השני וכך ניתן יהיה לבצע את אותה הפעולה ע״י העברת שני פרמטרים
לדוגמה

```
(lambda (x y) (* x y))
```

הופך ל..

```
(lambda (x) (lambda (y) (* x y)))
```

4. אי אפשר תמיד להמיר תוכנית מ־L2 ל־L23.
אם פונקציה מקבלת פונקציה אחרת כפרמטר (higher-order function), לא תמיד ניתן ליישם זאת ב־L23.

דוגמה:

```
(define apply-twice
  (lambda (f x)
    (f (f x))))
```

apply-twice מקבלת פונקציה f וערך x, ומפעילה פעמיים את f.
אם יודעים מראש את תוכן הפונקציה אפשר לשכתב זאת מבלי להעביר אותה אבל לא ניתן ליצור פונקציה כללית שמקבלת פונקציה ומפעילה אותה פעמיים

שאלה 2

<prim-op> ::= + | - | * | / | < | > | = | not | eq? | string=?
| cons | car | cdr | list | **dict | dict? | get** |pair? | list? | number?
| boolean? | symbol? | string?