1.1. המטרה של הפונקציה valueToLitExp היא להמיר ערכים מחושבים לביטויים.
ב- applicative-order, בפונקציית applyClosure, הארגומנטים (ביטויים) מחושבים לערכים עוד לפני פעולת ההצבה (פעולת ההצבה דורשת משתנים מסוג ביטוי בלבד – פונקציית substitute).
לכן אנו משתמשים בפונקציה valueToLitExp על מנת להמיר את הארגומנטים (ערכים) לביטויים.
כלומר, הפונקציה valueToLitExp פותרת את בעיית תאימות הטיפוסים.

1.2. הפונקציה valueToLitExp אינה רלוונטית ב- normal-order משום שבאסטרטגיה זו, הארגומנטים מחושבים כאשר הם נדרשים לכך. כלומר, בשלב זה של התוכנית, הארגומנטים אינם נדרשים לכך ולכן הם עוד בצורת ביטוי.

1.3. אסטרטגיה 1 לחישוב ביטוי מסוג let :

א. נחשב את הערכים ב- Bindings (משתנים מקומיים) עפ"י הסביבה הנוכחית.
ב. הגדרת המשתנים ב- Bindings כך שהם מקושרים לערכים שחושבו בשלב הקודם.
ג. הצבת ערכי ה- Bindings (המשתנים המקומיים) ב- Body (בגוף המבנה).
ד. חישוב של ה- Body לאחר ההצבות כך שהערך של ה- Body הוא בעצם הערך של הביטוי האחרון ב- Body.

אסטרטגיה 2 לחישוב ביטוי מסוג let :

משום שמבנה ה- let הוא בעצם קיצור תחבירי (syntactic-abbreviation), אנו ממירים את מבנה ה- let למבנה של הפעלת פרוצדורה (App-Exp) ולאחר מכן מחשבים את הביטוי של מבנה זה.

א. חישוב האופרטור.
ב. חישוב הפרמטרים.
ג. הפעלת האופרטור על הפרמטרים.

1.4.
א. חלוקה ב- 0 :

```
(\ 30 0)
```

ב. לולאה אינסופית :

```
(define loop (lambda () (loop)))
(define f (lambda (x) 5))
(f (loop))
```

ג. הפעלת אופרטור חישוב על אופרנדים בוליאניים :

```
(+ #t #f)
```

ד. שימוש במשתנה לא מוגדר :

```
(+ x 1)
```

1.5. ההבדל בין צורה מיוחדת לאופרטור פרימיטיבי הוא צורת החישוב של האופרטור ושל האופרנדים. באופרטור הפרימיטיבי – ראשית, מחשבים את כל האופרנדים ולאחר מכן מחשבים את הביטוי כולו. בעוד שהצורה המיוחדת מחשבת חלק/כל האופרנדים לבחירתה.

1.6. הסיבה המרכזית למעבר ממודל ההצבה למודל הסביבה היא שבמודל ההצבה, כל הפעלה של פרוצדורה כרוכה בשכתוב גוף המבנה שלה (חישוב ערכי הארגומנטים, שינוי שמות כל הפרמטרים של כל הפרוצדורות המוגדרות בגוף המבנה, החזרת ערכי הארגומנטים למבנה של

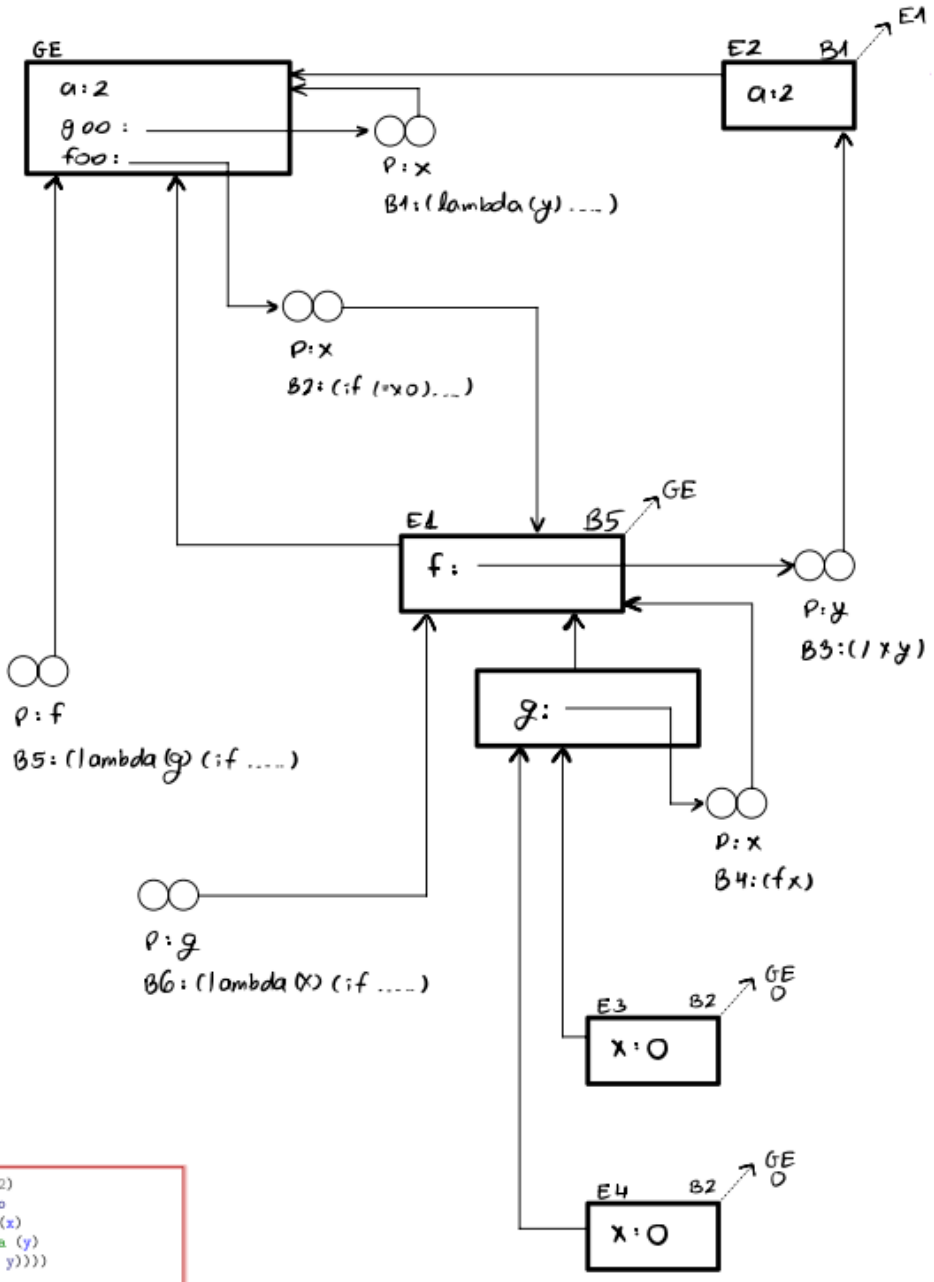
ביטויים והצבת הארגומנטים בכל VerRef מתאים). בעוד שמודל הסביבה מונע שכתוב מחדש של גוף המבנה של פרוצדורה באמצעות הרחבת הסביבה הגלובלית להיררכיה של פריימים ע"י הוספת פריים בכל הפעלה של פרוצדורה (שבו יוגדרו ה- Bindings של כל הפרמטרים של הפרוצדורה יחד עם הערכים שנשלחו עבורם). ועבור כל VerRef, מתבצע חיפוש של הערך שלו בהיררכית הפריימים – החל מהפריים האחרון ומעלה עד לסביבה הגלובלית.

דוגמה :

```
(define fact (lambda (n) (if (= n 0) 1 (* n (fact (- n 1)))) ))
(fact 4)
```

נשים לב שבמודל ההצבה, בפרוצדורות שהן רקורסיביות ישנה כמות גדולה מאד יחסית של הפעלות פרוצדורה הגוררות שכתוב מחדש בכל הפעלה – דבר שהוא מאד כבד. לכן במודל הסביבה – חלק נכבד מן הפעולות במודל ההצבה נחסך.

1.7. הסיבה המרכזית למימוש הסביבה באמצעות box היא תמיכה בעדכון\שינוי. לשם כך אנו עוטפים את הערך ב- "קופסה" באופן המאפשר גישה לערך ושינוי הערך. תמיכה זו מאפשרת שימוש יעיל ברקורסיה (letrec) ועיצוב הסביבה הגלובלית עם העברת משתנים גלובלים ופרוצדורות רקורסיביות הדדיות גלובליות.




```
(define a 2)
(define goo
  (lambda (x)
    (lambda (y)
      (/ x y))))

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

2.1.3. הביטוי ?bound הוא במבנה של צורה מיוחדת על מנת לאפשר גישה ל- Bindings של הסביבה הנוכחית. בנוסף לכך, בצורה מיוחדת זו אין צורך לחשב את הביטוי \ להוסיף Binding חדש לסביבה הנוכחית – דבר המתבצע באופרטור פרימיטיבי או בפונקציית משתמש.

2.2.3. הביטוי time צריך להיות במסגרת של צורה מיוחדת מכיוון שבאופן זה ישנה אפשרות למדוד בדיוק רב את משך זמן ה- evaluation של הפונקציה המתקבלת כפרמטר. מימוש הביטוי time בצורה פרימיטיבית היה גורר בנוסף מדידת משך הזמן של פעולת ה- parser.