Part 1 : Theoretical Questions

שאלה 1

כן, let צורה מיוחדת. כזכור, בעבודה הקודמת הגדרנו מה זה special form , וlet עונה על הגדרה זו. צורה זו נועדה לפתור את החיסרון של חישוב חוזר של הקריאה לפרוצדורה כלשהי שהוגדרה בתוכנית (בכל פעם שהביטוי מופיע). הצורה המיוחדת let מאפשרת להגדיר משתנים לוקאליים.
מבנה זה כולל 2 חלקים : הגדרת משתנים לוקליים (bindings) וגוף הקוד (body). הסמנטיקה של let היא בעצם לחשב את הbindings על פי הסביבה הקיימת, להגדיר את המשתנים ב-binding כך שהם מקושרים לערכים שחושבו ,הצבתם ב-body של הlet וחישוב הbody לאחר הצבתם. הערך של let הוא הערך של הביטוי האחרון ב-body.

שאלה 2

הפונקציה valueToLitExp נועדה על מנת להמיר מרשימת הערכים להצבה לרשימת ביטויים (LitExp) מקבילה. פונקציה זו נדרשת משיקולי תאימות טיפוסים, כלומר גוף הפרוצדורה להצבה מוגדר במושגים של CExp של הפארסר , אך הערכים להצבה הם כבר במושגים של Value (הערך) של האינטרפטר. מאחר והפונקציה substitute (מימוש ההצבה) מצפה לקבל מערך של CExp נצטרך לבצע המרה טיפוסים זו.

שאלה 3

הפונקציה valueToLitExp אינה נדרשת בnormal evaluation שכן אין צורך בהמרת האופרנדים בחזרה מערכים (value) לביטויים (CExp) כי הם אינם מחושבים בשיטה זו עדיין אלא רק בעת הפעלת הפרוצדורה.

שאלה 4

במודל הסביבות אנו נרצה להרחיב את הסביבה הנוכחית ע"י הוספת משתנים וערכיהם לסביבה הקיימת. בפונקציה applyClosure אנו מקבלים מערך של ערכים ועל מנת להרחיב את הסביבה הקיימת (makeExtEnv) אנו נדרשים לספק את שמות המשתנים החדשים וערכיהם (value) ולא ביטויים (CExp). לכן במודל הסביבות אין אנו צריכים את הפונקציה valueToLitExp הממירה את הערכים לביטויים.

שאלה 5

ישנם מספר סיבות מדוע עדיף להשתמש בnormal order ולא בapplicative order evaluation.
ראשית, כאשר יש בקטע הקוד if אז בשיטת normal לא נחשב את ערך המקרה (then\alt) בהתאם לtest) אותו איננו צריכים לחשב ולעומת זאת בapplicative נחשב גם את then וגם את alt על אף שבאחד מהם אין לנו צורך. כאשר הalt/then הם חישובים מורכבים לא יהיה יעיל לחשב את שניהם שלא לצורך (אלא מספיק את האחד הדרוש).
סיבה נוספת היא מניעת אפשרות לשגיאת זמן ריצה. לדוגמא:

```
(if (> 4 3)
    2
    (/ 4 0))
```

במקרה זה בשיטת הnormal יחזור הערך 2 שכן לא מתבצע חישוב של alt. לעומת זאת בשיטת applicative נקבל שגיאת זמן ריצה שכן לא ניתן לחלק 4 ב-0.

שאלה 6

נעדיף להשתמש בapplicative order evaluation על פני normal order כאשר אנו נדרשים לחשב את אותו ביטויי הקיים בפרוצדורה מספר פעמים. לדוגמא :

```
(
(lambda (y) (+ y y))
 (* 2 3)
)
```

בשיטת normal נחשב פעמיים את הביטוי (* 2 3) , לעומת זאת בשיטת applicative נחשב פעם אחת בלבד.

שאלה 7

a. נניח ויש ביטוי כלשהו שלא מכיל משתנים חופשיים. אם נבצע renaming אזי :

עבור כל משתנה x שהוא לא משתנה חופשי נשנה את שמו על ידי הוספת מספר. מאחר ואין משתנים חופשיים ערך הביטוי אינו משתנה כלל, כלומר זה שקול לביטוי שהיה לפני ה- renaming.

לכן אין צורך לשנות כי ערך הביטוי לא השתנה גם אחרי שינוי השם.

ניתן דוגמא להמחשה כפי שהוצג בתרגול :

(lambda (x) (+ x x)) ⟺ (lambda (x1) (+ x1 x1))

b. אלגוריתם naïve substitution יתבצע באופן הבא: (ללא renaming)

- ביצוע ההמרה של הארגומנטים ל- valueToLitExp (כפי שתואר בשאלה 2)
- הפעלת substitute על הLitExp על הארגומנטים עם שמות המשתנים וה-body שלהם בClosure.

שאלה 8