Assignment 5

1.1.b נוכיח שהפרוצדורה $append\$$ שקולה-CPS לפרוצדורה $append$.

נוכיח זאת באמצעות אינדוקציה בכך שלכל $n \in \mathbb{N}$ כך ש-$n = |lst1|$ ולכל $continuation$ אשר יסומן על ידי c מתקיים: $(append\$ \ lst1 \ lst2 \ c) = (c \ (append \ lst1 \ lst2))$.

<u>בסיס האינדוקציה:</u> $n = 0$

$$a - e\ [\ (append\$ \ `() \ lst2 \ c)\ ] \ \Rightarrow \ a - e[\ (c \ lst2)\ ] \ = \ a - e[\ (c \ (append \ `() \ lst2))\ ]$$

<u>הנחת האינדוקציה:</u> עבור $n = k \in \mathbb{N}$ הטענה מתקיימת לכל $k \geq i$. כלומר

$$(append\$ \ lst1 \ lst2 \ c) = \ \big(c \ (append \ lst1 \ lst2)\big) \ where \ |lst1| = i \leq k$$

<u>צעד האינדוקציה:</u> יהא $n = k + 1, k \in \mathbb{N}$, אזי:

$$a - e\ [\ (append\$ \ lst1 \ lst2 \ c)\ ] \ \Rightarrow$$

$$a - e\ [\ (append\$ \ (cdr \ lst1) \ lst2 \ (lambda \ (res) \ (c \ (cons \ (car \ lst1) \ res))))\ ] \ \Rightarrow$$

מהנחת האינדוקציה, נקבל:

$$a - e\ [\ ((lambda \ (res) \ (c \ (cons \ (car \ lst1) \ res))) \ (append \ (cdr \ lst1) \ lst2))\ ] \ \Rightarrow$$

$$a - e\ [\ (c \ (cons \ (car \ lst1) \ (append \ (cdr \ lst1) \ lst2)))\ ] \ =$$

$$a - e\ [\ (c \ (append \ lst1 \ lst2))\ ]$$

2.d

<u>$reduce1 - lzl$:</u> נשים לב כי הפרוצדורה $reduce1 - lzl$ מחזירה את הערך המצטבר של כלל אברי הרשימה הנתונה, לכן נרצה להשתמש בפרוצדורה זו כאשר נתונה לנו רשימה סופית (אם היא אינסופית התוכנית לא תסתיים) וכאשר אנחנו מעוניינים לקבל את הערך המצטבר של כלל אברי הרשימה.

<u>$reduce2 - lzl$:</u> נשים לב כי הפרוצדורה $reduce2 - lzl$ מחזירה את הערך המצטבר של n האיברים הראשונים של הרשימה הנתונה, לכן נרצה להשתמש בפרוצדורה זו כאשר אנו מעוניינים לדעת מה הוא הערך המצטבר של אברי הרשימה רק עד אינדקס מסוים, או שבמידה והרשימה הנתונה לנו היא אינסופית אז לא יהיה ניתן להשתמש ב-$reduce1 - lzl$ (כי התוכנית לא תסתיים במקרה וזה) ונצטרך להשתמש ב$reduce2 - lzl$.

<u>$reduce3 - lzl$:</u> נשים לב כי הפרוצדורה $reduce3 - lzl$ מחזירה לנו רשימה אשר האיבר ה-i שלה מייצג את הערך המצטבר של i האיברים הראשונים של רשימת הקלט, לכן נרצה להשתמש בפרוצדורה זו כאשר נרצה לראות את מגמת ההתקדמות של הערך המצטבר או שבאופן כללי נרצה לקבל תמונת מצב מפורטת יותר על הערכים המצטברים של הרשימה הנתונה כך שבמקום השימוש ב$reduce1 - lzl$ ו$reduce2 - lzl$ בהן יכלנו לקבל ערך מצטבר יחיד, פה נוכל לקבל ערך מצטבר לכל n טבעי על ידי הפעלת הפרוצדורה בפעם אחת בלבד.

2.g

<u>יתרון של $generate - pi - approximations$ על פני $pi - sum$:</u>

נניח ונרצה לחשב קירוב של π ביחס למספר קירובים, אז במימוש של $pi - sum$ כל פעם שנרצה קירוב כלשהו נצטרך להתחיל את החישוב מההתחלה וזאת בניגוד למימוש של

$generate - pi - approximations$ שבו פשוט יהיה ניתן לייצר קירובים נוספים תחת אותה הרשימה העצלה אשר מוחזרת כפלט.

<u>חסרון של $generate - pi - approximations$ על פני $pi - sum$:</u>

בכך שבמימוש של $generate - pi - approximations$ אנו משתמשים ב-$lazy\ list$ בניגוד למימוש של $pi - sum$ בו מוחזר ערך יחיד, אם נרצה לקבל קירוב ספציפי לפאי אז במימוש של $pi - sum$ נוכל לקבל אותו קירוב ספציפי על הפרמטרים שנכניס לפונקציה. לעומת זאת, במימוש של
$generate - pi - approximations$ אם נרצה קירוב ספציפי לפאי אז בשביל לגשת לאותו קירוב נצטרך לייצר רשימה אשר תכיל גם את כל הקירובים שפחות מדויקים לפאי אשר יהיו לפניו ברשימה העצלה שלא בהכרח רצינו לקבל.

a.3.1 נסמן $A = t(s(s), G, s, p, t(K), s),\ \ B = t(s(G), G, s, p, t(K), U)$ ונבצע את האלגוריתם:

1. $S = \{s = G\}$

$A \circ S = t(s(s), s, s, p, t(K), s)$

$B \circ S = t(s(s), s, s, p, t(K), U)$

2. $S = \{s = G, s = U\}$

$A \circ S = t(s(s), s, s, p, t(K), s)$

$B \circ S = t(s(s), s, s, p, t(K), s)$

לכן הMGU הוא $S = \{s = G, s = U\}$.

b.3.1 נסמן $A = p([v \mid [V \mid W]]),\ \ B = p([[v \mid V] \mid W])$ ונבצע את האלגוריתם:
בהצבה הראשונה אנו מקבלים השוואה בין האיבר הראשון ברשימה ב-A לאיבר הראשון ברשימה ב-B כאשר האיבר הראשון ברשימה ב-A הוא סמל v והאיבר הראשון ב-B הוא רשימה $[v\ |V]$, כלומר הם לא מאותו מבנה ולכן נקבל כישלון.

פתרון עץ ההוכחה בעמוד הבא

3.3

natural_number(zero). %n1
natural_number(s(X)) :- natural_number(X). %n2
plus(X, zero, X) :- natural_number(X). %p1
plus(X, s(Y), s(Z)) :- plus(X, Y, Z). %p2
?- plus(s(s(zero)), s(X), s(s(s(s(zero))))).