שאלה 1: Lazy Lists

1. תהיינה list1 ו-list2 שתי רשימות (לא עצלות) כלשהן.  
   נגדיר ש- list1 שקולה ל-list2 (וגם ההפך, זהו יחס שקילות) אם מתקיים אחד מהדברים הבאים:
   1. list1=list2='()
   2. (car list1)=(car list2)  
      וגם (cdr list2) שקול ל-(cdr list1)

ונסמן list1=list2.  
  
נגדיר ש-lzl1 שקולה ל-lzl2 (וגם ההפך, זהו יחס שקילות) אם לכל מתקיים  
ש-(take lzl1 n) שקול ל-(take lzl2 n)ונסמן lzl1=lzl2.

1. טענה: even-square-1 שקול ל-even-square-2.  
   כלומר לכל נוכיח כי (take even-square-1 n)  
   שקול ל-(take even-square-2 n).

נשים לב ש:

(head even-square-1)=(head even-square-2)=0

בפרט even-square-1 ו-even-square-2 לא ריקות.

תהי ב-take-from(lzl, n) פרוצדורה הטיפוס [LzL\*Number -> LzL] אשר מקבלת רשימה עצלה ומספר טבעי כך ש-n < length(lzl) ומחזירה רשימה עצלה של כל האיברים של lzl החלק מהאיבר ה--י של lzl (כולל) (המספור מתחיל ב-0).  
נשים לב לכך ש:

* (take-from lzl 0)=lzl
* (take-from lzl 1)=(tail lzl)

נוכיח שלכל מתקיים ש-(take-from even-sqaure-1 n) שקול  
ל-(take-from even-sqaure-2 n).

נוכיח באינדוקציה על .  
בסיס: : אז מתקיים

(take even-square-1 0) ==>\* '()  
(take even-square-2 0) ==>\* '()

הנחה: לכל הטענה נכונה.  
צעד: אז .  
נשים לב ש:

(take even-square-1 n) ==>\*  
(cons (head even-square-1)  
 (take (tail even-square-1) (- n 1)))

וגם:

(take even-square-2 n) ==>\*  
(cons (head even-square-2)  
 (take (tail even-square-2) (- n 1)))

מהנחת האינדוקציה נקבל ש:

(take (tail even-square-1) (- n 1))  
שקול ל-(take (tail even-square-2) (- n 1)))

לכן:

(cons (head even-square-1)  
 (take (tail even-square-1) (- n 1)))=  
(cons (head even-square-2)  
 (take (tail even-sqaure-2) (- n 1)))=  
(take even-square-2 n)

ד

שאלה 2: CPS

1. תהי f פרוצדורה מטיפוס [T1\*...\*Tn->T1|T2] כך ש-f מחזירה ערך מטיפוס T1 במקרה "הצלחה" וערך מסוג T2 במקרה של "כישלון".  
   נגדיר ש-f שקולה לגרסה ה-Success-Fail-Continuations שלה המסומנת f$ אם לכל קלט x1,...,xn ולכל פרוצדורה success-cont (מטיפוס [T1->Tsc] כך ש-Tsc יכול להיות כל טיפוס) מתקיים:  
   אם (f x1 ... xn) מניב "כישלון", אז קיימת פרוצדורה fail-cont כך שקיימים ערכים y1,...,ym כך ש-

(f$ x1 ... xn success-cont fail-cont)  
=(fail-cont y1 ... ym)=(f x1 ... xn)

כאשר הטיפוס של fail-cont הוא [T1\*...\*Tm->T2]  
או שקיימת פרוצדורה fail-cont מטיפוס [Empty->T2] כך ש-

(f$ x1 ... xn success-cont fail-cont)=(fail-cont)  
=(f x1 ... xn).

אחרת, במקרה של "הצלחה", לכל פרוצדורה fail-cont מתקיים:

(f$ x1 ... xn success-cont fail-cont)  
=(success-cont (f x1 ... xn)).

הערות:

* + המקרים "הצלחה" ו-"כישלון" של f אינם מוגדרים. על מי שמגדיר את הפונציה f, להגדיר מהם עבור f.
  + על fail-cont להיות פרוצדורה המחזירה T2. מספר הארגומנטים וטיפוסם אינו מוגדר. על מי שמגדיר את הפונקציה f, להגדיר את ממשק הפונקציה fail-cont.

1. במקרה שלנו, מקרה ההצלחה הוא כאשר קיים זוג ברשימה אשר הערך הראשון שלו שווה ל-key, ומקרה הכישלון הוא אחרת.

טיפוס הפרוצדורה get-values$ הוא  
[List<Pair<Symbol,T>>\*Symbol\*[T->T1]\*[Empty->T2] -> T1|T2].  
(מכאן שטיפוס הפונקציה fail-cont הוא [Empty->T2]).  
  
טענה: הפרוצדורה get-value$ שקולה לפרוצדורה get-value.  
כלומר לכל פרוצדורה success-cont מטיפוס [T->T1], לכל list מהטיפוס List<Pair<Symbol,T>> ולכל key מטיפוס Symbol:  
אם (get-value list key) מניב 'fail, אז קיימת פונקציה fail-cont (מהטיפוס [Empty->'fail] כך ש-

(get-value list key)  
=(fail-cont)=(get-value$ list key success-cont fail-cont).

אחרת (במקרה של הצלחה), אז לכל פרוצדורה fail-cont מטיפוס [Empty->T2] מתקיים:

(get-value$ list key success-cont fail-cont)  
=(success-cont (get-value list key)).

תהיו list רשימה מטיפוס List<Pair<Symbol,T>>, מפתח key מטיפוס Symbol  
ו-success-cont פרוצדורה מטיפוס [T->T1].  
הוכחה האינדוקציה על אורך הרשימה list: נסמן .  
בסיס:   
במקרה זה, הרשימה ריקה, כלומר list='().  
לכן, עבור הפרוצדורה fail-cont המוגדרת להיות (lambda () 'fail) נקבל:

(get-value$ list key success-cont fail-cont) ==>\*  
(fail-cont)='fail=(get-value list key) .

הנחה: עבור כל , הטענה נכונה.

צעד: אז , כלומר הרשימה אינה ריקה.  
*יהי* '(k . v) *הזוג הראשון ברשימה כלומר* '(k . v) = (car list)*.  
נחלק למקרים:*

מקרה א' – : אז (get-value list key)=v ולכל פרוצדורה  
fail-cont מטיפוס [Empty->T2]:

(get-values list key) ==>\* (cdr (car list))  
=(car '(k . v))=v

וגם:

(get-value$ list key success-cont fail-cont)  
==>\* (success-cont (cdr (car list)))  
= (success-cont (cdr '(k . v))) = (success-cont v)  
= (success-cont (get-value list key))

מקרה ב' – :

(get-value$ list key success-cont fail-cont)  
==>\* (get-value$ (cdr list) key success-cont fail-cont))

וגם:

(get-values list key) ==>\* (get-values (cdr list) key)

נשים לב שאורך הרשימה (cdr list) קטן ממש מ-. ולכן מהנחת האינדוקציה נקבל:

אם (get-values (cdr list) key)='fail, אז נקבל שקיימת פרוצדורה  
fail-cont מטיפוס [Empty->T2] כך ש:

(get-value$ list key success-cont fail-cont)  
==>\* (get-values$ (cdr list) key success-cont fail-cont)  
=(fail-cont)=(get-values (cdr list) key)=(get-values list key).

אחרת, (get-values (cdr list) key) הניב הצלחה.  
תהי fail-cont פרוצדורה מטיפוס [Empty->T2] כלשהי.  
אז נקבל (מהנחת האינדוקציה) ש:

(get-value$ list key success-cont fail-cont)  
==>\* (get-values$ (cdr list) key success-cont fail-cont)  
=(success-cont (get-values (cdr list) key))  
=(success-cont (get-values list key))