שאלה 1:

סעיף ב: נוכיח ש (pipe$ lst-fun cout)=(cout(pipe lst-fun))

נוכיח באינדוקציה על גודל ה lst-fun כלומר $|lst-fun| = n$ כלומר נוכיח באינדוקציה על n

בסיס האינדוקציה: n=1

a-e[(pipe$ (lst-fun cout))] ➔a-e[(cout (car lst-fun))] ➔a-e [cout (pipe lst-fun)]

הנחת האינדוקציה: נניח עבור כל $|lst-fun| = k \leq n$ מתקיים ש

(pipe$ lst-fun cout)=(cout(pipe lst-fun))

צעד : נוכיח עבור $|lst-fun| = n + 1$

- נניח בה"כ שהפונקציה החדשה שהוספנו שגרמה ל n+1 נמצאת ראשונה

a-e [(pipe$ (lst-fun cout))]➔a-e[(pipe$ (cdr lst-fun) (lambda(cdr-res) (compose$ (car lst-fun) cdr-res cout)]➔*a-e[((lambda(cdr-res)(compose$ (car lst-fun) cdr-res cout))(pipe (cdr lst-fun)))➔

a-e[(compose$(car lst-fun) pipe(cdr lsr-fun) cout)]➔**a-e[(cout(compose$ (car lst-fun) pipe(cdr lst-fun)]➔a-e[(cout(pipe(lst-fun))]

בנדרש.

עכשיו נברר את *➔ ואת **➔

1) *➔: מהנחת האינדוקציה מתקיים את השלב הבא
2) **➔: נוכיח עכשיו את השוויון (compose$ f$ g$ coun)=(coun(compose f g))
3) ** אנו יכולים להניח ש (cout f)=(f$ cout) וגם (cout g)=(g$ cout)

a-e[(compose$(f$ g$ coun))]➔ a-e[(coun(lambda(x coun)(f$x (lambda(x-res) (g$ x-res coun)))))]➔ a-e(cont(lambda (x cont)(f$ x (lambda (res)(cont (g res))))➔ a-e[(cout(lambda (x cout)(lambda(res)(cout (g res))))(f x)]➔ a-e[(cout(lambda(x cout2)(cout2 (g(f x)))))];;we can see that compose it not a recursive for that we get that cout2 is the function id ➔a-e[(cout(lmbda(x)(g(f x))))]➔a-e[(cout(compose f g))]

שאלה 2:

סעיף א:

Lzl1 שווה ל lzl2 אם מתקיים שעבור הערכים הראשונים (יתכן יותר מאיבר אחד שהוא מוגדר) הם שווים ,כלומר לכל val1 שמוגדר ב Lzl1 קיים איבר val2 באיברים המוגדרים ב lzl2 כך ש val1=val2 , ועבור פונקציית המשיך של lz2 ושל lzl1 מחזירות אותו דבר כלומר או שהן מקבלות אותו סוג של ערכים ואותו ערכים ומחזירות אותו תשובה או שאם אחד מחזירה שגיה אז השנייה מחזירה שגיה

סעיף ב:

נוכיח באינדוקציה על מספר הקריאות K

מקרה בסיס : k=0

לפי הגדרת fib1 האיבר הראשון הוא 0 ולפי הגדרת fib2 האיבר הראשון הוא 0 לכן מתקיים שבקריאה הראשונה לערכים המוגדרים הן שווים כלומר

(head fib1)=0=(head fib2)

<u>הנחת האינדוקציה</u>: נניח עבור כל קריאה k≤n מתקיים ש (head fibs1)=(head fibs2) בקראה ה K

<u>צעד האינדוקציה</u> : נוכיח עבור k=n+1

נניח בלי בה"כ ש הערכים a,b הם הערכים של הקראיות n-1,n-2 (לפי הנחת האינדוקציה מתקיים ש לכל קריאה לערכים שהם בקריאות שקטנות מ n מתקיים את השוויון לכן נקח את האחרונים) , נוכיח עבשיו שהערכים X שמחזירה fibs1 בקריאה ה n הוא אותו ערך y שמחזירה fibs2 בקריאה ה n ,כלומר נוכיח ש x=y .

מהגדרת fibs1 מתקיים מכיוון ש X נוצר בקריאה ה n ולפי הגדרת fibs1 מתקיים ש x=a+b כך ש a נוצר בקראה ה n-1 ו b נוצר בקריאה ה n-2

מהגדרת fibs2 אנו קוראים לפונקציה ((lz-lst-add (tail lz1) (tail lz2 שמחזירה רשימה עצלה שמכילה בראש את הסכום של הראש של הרשימה lz1 והראש של lzl2 כאשר lzl1 היא רשימה עצלה כך ש בראש שלה יש את הערך z1 שנוצר בקראה n-1 וגם lzl2 היא רשימה עצלה כך ש בראש שלה יש את הערך z2 שנוצר בקראה n-2 כלומר y=z1+z2 ולפי הנחת האינדוקציה מתקיים ש z1=a,z2=b כלומר מתקיים ש x=a+b=z1+b=z1+z2=y כנדרש

שאלה 3:

<u>סעיף 3.1:</u>

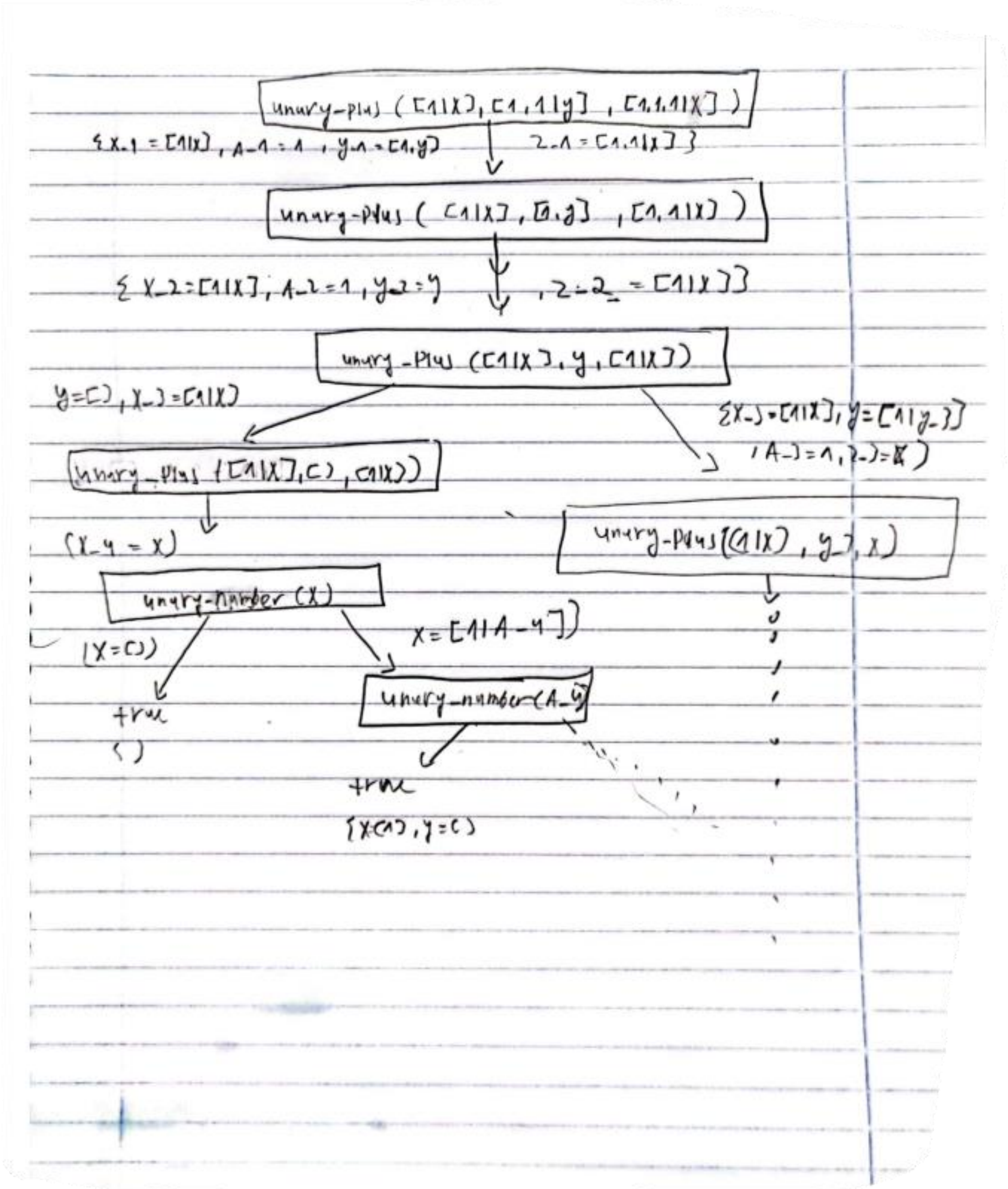
1) unify[p(v(v(d(M),M,ntuf3),X)), p(v(d(B),v(B,ntuf3),KtM))]
   initially: s={}, A=p(v(v(d(M),M,ntuf3),X)),B=p(v(d(B),v(B,ntuf3),KtM))
   s=so$\{\mathrm{v}(\mathrm{d}(\mathrm{M}),\mathrm{M},\mathrm{ntuf3})=\mathrm{v}(\mathrm{B},\mathrm{nutf3})\}$=> illegal substitution
   Answer: No such substitution because on the left there is 2 v get two args but on the right side it get 3

2) unify[n(d(D),D,d,k,n(N),K),n(d(d),D,d,k,n(N),d)]
   initially: s={}, A=n(d(D),D,d,k,n(N),K),B=n(d(d),D,d,k,n(N),d)
   1) s=so{D=d}={D=d}
      Aos=n(d(d),d,d,k,n(N),K)
      Bos=n(d(d),d,d,k,n(N),d)

   2) s=so{K=d}={D=d,K=d}
      Aos=n(d(d),d,d,k,n(N),d)
      Bos=n(d(d),d,d,k,n(N),d)
      Answer:s={D=d,K=d}

__סעיף 3.3:__

סעיף א:



סעיף ב: לפי הגדרת עץ הצלה שהגדרנו בהרצאה ,הוכחנו שאם קיים לפחות מסלול הצלחה בעץ אז זה עץ הצלחה ולפי הציור שצירנו אפשר לראות שקיים לפחות מסלול הצלחה 1 לכן זו עץ הצלחה

סעיף ג: לפי הציור אפשר לראות שזה עץ הצלחה אין סופי

סעיף ד: ניתן הוכיח את השאלתה הזו באמצעות ה programs וזה נותן לנו אין סוף פתרונות מכיוון שזה עץ אין סופי

סעיף ה: לא כי יתכן ש cons יהיה פונקציה רקורסיבית לכן יתכן שיש כמה מקרים שהם לא יסתיימו לכן L9 לא תהיה rational logic programming language דוגמה לכך ש L9 תהיה רקורסיבית (cons 14(cons 13(cons 12) (cons…….)))))..)