עידן דויטש
u.multinet.co.il

מודלים חישוביים – פתרון תרגיל 2

1. הוכח את הגרסה היותר כללית למשפט רייס בו סמנטיקה מוגדרת להיות שקילות בשפה. כלומר אם L(p1)=L(p2) אז P(p1)=P(p2).

הוכחה:
נחלק את הבעיות הלא-טריוויאליות והסמנטיות ל-2 סוגים: סוג א' וסוג ב'. בעיות מסוג א' הן בעיות שמחזירות F עבור התוכנית שמתבדרת לכל קלט (loopy). ובעיות מסוג ב' הן בעיות שמחזירות T עבור התוכנית שמתבדרת לכל קלט (loopy). תהי P לא טריוויאלית וסמנטית ונניח בשלילה שP כן כריעה ונראה רדוקציה $halt_0 \le P$.

<u>אם P היא בעיה מסוג א</u>:
1. P(loopy) = F.
2. קיימת תוכנית yes כך ש- P(yes) = T (כי P אינה טריוויאלית).

נגדיר את הרדוקציה $halt_0 \le P$:

$$halt_0(f) := P(g) \qquad where \qquad g = \lambda y.\begin{cases} yes(y) & f() = f() \\ yes(y) & otherwise \end{cases}$$

<u>אם P היא בעיה מסוג ב</u>:
1. P(loopy) = T.
2. קיימת תוכנית no כך ש- P(no) = F (כי P אינה טריוויאלית).

נגדיר את הרדוקציה $halt_0 \le P$:

$$halt_0(f) := not(P(g) \qquad where \qquad g = \lambda y.\begin{cases} no(y) & f() = f() \\ no(y) & otherwise \end{cases})$$

הוכחת נכונות:

<u>התוכנית עוצרת</u>: הנחנו ש-P היא בעיה כריעה, לכן עבור כל תוכנית יוחזר T או F וערך זה יועבר ל$halt_0$ ומכאן שבכל מקרה התוכנית תעצור.

<u>הערך המוחזר נכון</u>:

עבור בעיות מסוג א' – אם f עוצרת, אזי הביטוי f()=f() ניתן לחישוב (כלומר אינו נכנס ללולאה אינסופית) ויחזיר T. במקרה זה, עבור כל y, התוכנית g תתנהג באותה הצורה כמו yes ובפרט, תחזיר T עבור אותם ערכים ש-yes מחזירה עבורם T (ורק עבורם). כלומר g שקולה ל-yes (השפות של שתיהן שקולות). היות ו-P סמנטית, P(g)=P(yes)=T ולכן $halt_0$ יחזיר T כנדרש. אם f אינה עוצרת, אז לכל קלט y שיועבר ל-g, התוכנית g תכנס ללולאה אינסופית (כי חישוב הביטוי f()=f() יכנס ללולאה אינסופית) ומכאן ש-g לא תחזיר T עבור אף ערך. כלומר, g שקולה לתוכנית loopy (השפות שלהן שקולות). היות ו-P סמנטית P(g)=P(loopy)=F. כלומר, $halt_0$ יחזיר F כנדרש.

עבור בעיות מסוג ב' – אם f עוצרת, אזי הביטוי f()=f() ניתן לחישוב (כלומר אינו נכנס ללולאה אינסופית) ויחזיר T. במקרה זה, עבור כל y, התוכנית g תתנהג באותה הצורה כמו no ובפרט, תחזיר T עבור אותם ערכים ש-no מחזירה עבורם T (ורק עבורם). כלומר g שקולה ל-no (השפות של שתיהן שקולות). היות ו-P סמנטית, P(g)=P(no)=F ולכן לאחר not יוחזר מ-$halt_0$ הערך T כנדרש. אם f אינה עוצרת, אז לכל קלט y שיועבר ל-g, התוכנית g תכנס ללולאה אינסופית (כי חישוב הביטוי f()=f() יכנס ללולאה אינסופית) ומכאן ש-g לא תחזיר T עבור אף ערך. כלומר, g שקולה לתוכנית loopy (השפות שלהן שקולות). היות ו-P סמנטית P(g)=P(loopy)=F ולאחר not יוחזר מ- $halt_0$ הערך F כנדרש.

סה"כ הראנו רדוקציה $halt_0 \le P$. כיוון ש-$halt_0$ אינה חשיבה, נסיק שגם P אינה חשיבה.

.2

a. כתוב רדוקציית מיפוי Sort $\leq_m$ --Halt.

פיתרון:

i. **טענה:** הפונקציה Sort היא פונקציה חשיבה.

הוכחה:
יהי l רשימה של מספרים. נכתוב פונקציה שמקבלת את l ועוברת איבר איבר ברשימה החל מתחילתה. בכל שלב, התוכנית תשווה בין 2 איברים עוקבים. התוכנית תמשיך לרוץ כל עוד $l_i \leq l_{i+1}$ או עד שנגיע לסוף הרשימה. אם במהלך הבדיקה נגלה כי קיים i עבורו $l_i >= l_{i+1}$ נחזיר F. אם הגענו לסוף הרשימה מבלי לעצור, נחזיר T.

הוכחת נכונות:
<u>התוכנית תעצור</u>: הרשימה היא סופית ולכן בכל מקרה או שנעצור מתישהו במהלך סריקת הרשימה (כי היא לא מסודרת) או שנעצור בסופה (כאשר נגיע לסוף הרשימה).
<u>הערך המחזר נכון</u>: ע"פ ההגדרה, אם הרשימה ממוינת לכל איבר ברשימה מתקיים $l_i \leq l_{i+1}$. אם מצאנו 2 איברים עוקבים להם לא מתקיים תנאי זה – הרי שהרשימה אינה ממוינת. אם הגענו לסוף הרשימה מבלי להפר את התנאי, כל האיברים ממוינים כראוי ע"פ טרנזיטיביות של $\leq$.

ii. **נראה רדוקציית מיפוי**

$$halt--(p,x) := Sort(f(p,x)) \quad where \quad f = \lambda p,x.\begin{cases} list(5,6,7) & p(x)=T \\ list(5,2,7) & otherwise \end{cases}$$

הוכחת נכונות:
<u>התוכנית עוצרת</u>: f מוגדרת ע"י הפרדיקט p והקלט x. p הוא פרדיקט בשפה --c וככל תוכנית בשפה זו, הוא חייב לעצור. לכן p יעצור ויחזיר ערך T או F. בכל מקרה, התוכנית f לאחר מכן תחזיר רשימה (מסודרת או לא) ובפרט תעצור.

<u>הערך המוחזר נכון</u>: f מקבלת את הפרדיקט p והקלט x. כיוון שp בהכרח עוצרת, הפלט המוחזר מ-p(x) הוא בהכרח T או F. f בודקת איזה ערך מחזיר p(x). אם מוחזר T, f תחזיר רשימה מסודרת (שתועבר ל-sort אשר יחזיר T כנדרש). אם מוחזר F, f תחזיר רשימה שאינה ממוינת (שתועבר ל-sort אשר יחזיר F כנדרש).

מסקנות מרדוקציית המיפוי
כיוון שf חשיבה וכן כיוון שSort חשיבה, רדוקציית המיפוי שהראנו מוכיחה כי גם --halt חשיבה.

b. הוכחת הנכונות שלנו ברדוקציית המיפוי ל---halt מסתמכת על כך שהפונקציה f עוצרת. f בוודאות עוצרת כיוון ש-p היא תוכנית הכתובה בשפה --c ולכן חייבת לעצור. בשפות אחרות כמו ++c, אין הכרח ש-p תעצור ולכן f אינה תעצור בוודאות. בשל כך, לא ניתן לבצע רדוקצייה ל-halt אך כן ניתן לבצע לפונקציה --halt.

c. כתוב רדוקציית מיפוי Halt $\leq_m$ A כאשר A שפה רקורסיבית (כריעה).

פתרון:
A היא שפה כריעה ולכן קיים פרדיקט p המכריע אותה. נגדיר:

$$A(x) := Halt(f(x)) \quad where \quad f = \lambda x.\begin{cases} < \lambda x.T, 1 > & p(x)=T \\ < \lambda x.loopy(x), 1 > & otherwise \end{cases}$$

הוכחת נכונות:

<u>התוכנית עוצרת</u>: A היא שפה כריעה ולכן הפרדיקט שמכריע את השפה יחזיר T או F (ובוודאות יעצור). התוכנית f משתמשת בפרדיקט זה ובהתאם לערך שהוא מחזיר מחזירה זוג סדור של פונקציה וערך. אין הרצה נוספת ב-f ולכן f בוודאות תעצור.

<u>הערך המוחזר נכון</u>: אם x שייך לשפה A, הפרדיקט p המכריע את A חייב להחזיר בעבורו T. לכן f תחזיר את הזוג הסדור של התוכנית שתמיד מחזירה T והערך 1. תוכנית זו עוצרת על כל ערך ובפרט על הערך 1 ולכן כאשר תועבר לHalt יוחזר T כנדרש.
אם x אינו שייך לשפה A, הפרדיקט p המכריע את A חייב להחזיר בעבורו F. לכן f תחזיר את הזוג הסדור של התוכנית loopy שאינה עוצרת והערך 1. תוכנית זו אינה עוצרת על אף ערך ובפרט אינה עוצרת עבור הערך 1 ולכן כאשר תועבר לHalt יוחזר F כנדרש.

d. הוכח או הפרך: אם קיימת רדוקציה A≤B אז קיימת גם רדוקציית מיפוי $A \le_m B$.

פיתרון: <u>הטענה אינה נכונה</u>.

הוכחה:
נגדיר את השפה diverge המכילה את כל הזוגות הסדורים <p.x> כך שp(x) מתבדרת.
נשים לב כי למעשה diverge היא משלים לתוכנית halt. בין התוכניות קיימת רדוקציה halt ≤ diverge המוגדרת להיות halt(p,x):=not(diverge(p,x)). נכונות הרדוקציה נובעת ישירות מכך שdiverge הוא משלים של halt (נשים לב כי halt כריעה למחצה).

נניח בשלילה שהטענה כן נכונה. כלומר, קיימת גם רדוקציית מיפוי $halt \le_m diverge$. לכן, ע"פ מה שהוכחנו בכיתה קיימת גם רדוקצית המיפוי של המשלימים $\overline{diverge} \le_m \overline{halt}$.
הראנו בכיתה ש $\overline{halt} \notin RE$ ולכן, ע"פ רדוקציית המיפוי של המשלימים גם $\overline{diverge} \notin RE/R$. אבל $\overline{diverge} = halt$ וכן מתקיים $halt \in RE/R$. קיבלנו סתירה ולכן ההנחה בשלילה אינה נכונה. כלומר אם קיימת רדוקציה בין 2 שפות, אין הכרח שקיימת גם רדוקצית מיפוי בינהן.

3. הוכח או הפרך: הפונקציה $s: N \to N$ חשיבה.
$s(n) = \max\{f_i(j) : i, j \le n\} + 1$ כאשר $f_0, f_1 ...$ הוא סידור של כל הפונקציות המספריות החשיבות.

פיתרון:
הטענה אינה נכונה. נניח בשלילה ש-s חשיבה. היות ו-s חשיבה, היא תופיע בסידור הפונקציות המספריות החשיבות $f_0, f_1 ...$. נניח שמיקומה של s בסידור הוא t.
ניתן להסתכל על קבוצת הפלטים $\{f_i(j) : i, j \le n\}$ כמטריצה A בגודל nXn כאשר השורות מסמלות את הפונקציות $f_0, f_1 ... f_{n-1}$ והעמודות מסמלות קלטים אפשריים (כלומר את המספרים הטבעיים 1...n). נשים לב כי הפונקציה s בוחרת למעשה את הערך המקסימאלי במטריצה זו ומוסיפה לו 1.

לכל n>t, הפונקציה s נמצאת בשורה הt+1 של הפונקציה (מספור הפונקציות מתחיל מ-0).
כלומר ערך הפונקציה s(n) הוא $A_{t+1,n}$. ע"פ ההגדרה של s, $A_{t+1,n}$ הוא הערך המקסימאלי במטריצה. אך s מוסיפה לערך זה 1, כלומר קיבלנו כי $s(n) = A_{t+1,n} + 1 \Rightarrow s(n)=s(n)+1$.
זו כמובן סתירה ולכן הטענה המקורית אינה נכונה.

4. הוכח שהמחלקה R סגורה תחת הפעולות הבאות (ניתן להשתמש במילים במקום בקוד):

a. **איחוד**
נראה שהשפה $L_1 \cup L_2$ כריעה ע"י הגדרת פרדיקט שיכריע אותה:
$$p(c) = p_1(c) \vee p_2(c)$$
כאשר p_1, p_2 הם פרדיקטים המכריעים את השפות L_1, L_2 בהתאמה.

הוכחת נכונות:
L_1, L_2 הן שפות כריעות. לכן, קיימים p_1, p_2 פרדיקטים, כך שלכל $c \in L_1, p_1(c) = T$ אחרת $p_1(c) = F$ וכן לכל $c \in L_2, p_2(c) = T$ אחרת $p_2(c) = F$. יהי $c \in L_1 \cup L_2$, אזי $c \in L_1$ או $c \in L_2$ (או לשניהם כמובן). אם $c \in L_1$ אז $p_1(c) = T$ ואם $c \in L_2$ אז $p_2(c) = T$. כלומר, אם $c \in L_1 \cup L_2$ על

אחד הפרדיקטים לפחות להחזיר T. לכן, $p(c)=p_1(c)\vee p_2(c)=T$.
אם $c\notin L_1\cup L_2$ אזי $c\notin L_1$ או $c\notin L_2$ ולכן $p_1(c)=F$ וגם $p_2(c)=F$. לכן, $p(c)=p_1(c)\vee p_2(c)=F\vee F=F$ כנדרש.

b. **חיתוך**
נראה שהשפה $L_1\cap L_2$ כריעה ע"י הגדרת פרדיקט שיכריע אותה:

$$p(c)=p_1(c)\wedge p_2(c)$$

כאשר p_1,p_2 הם פרדיקטים המכריעים את השפות L_1,L_2 בהתאמה.

הוכחת נכונות:
L_1,L_2 הן שפות כריעות. לכן, קיימים p_1,p_2 פרדיקטים, כך שלכל $c\in L_1, p_1(c)=T$ אחרת $p_1(c)=F$ וכן לכל $c\in L_2, p_2(c)=T$ אחרת $p_2(c)=F$. יהי $c\in L_1\cap L_2$, אזי $c\in L_1$ וגם $c\in L_2$. לכן $p(c)=p_1(c)\wedge p_2(c)=T\wedge T=T$. אם $c\notin L_1\cap L_2$ אזי $c\notin L_1$ או $c\notin L_2$ (או שאינו שייך לשניהם). לכן, $p_1(c)=F$ או $p_2(c)=F$ (או שניהם). ומכאן שגם, $p(c)=p_1(c)\wedge p_2(c)=F$ כנדרש.

c. **שרשור**
נראה שR סגור גם עבור שירשור, ע"י הגדרת פרדיקט שיכריע שרשור של 2 שפות:

```
p(c)
{
  len = length(c);

  for (int i=0; i<= len; i++)
  {
    if( p1(subStr(c,0,i)) &&  p2(subStr(c,i,len))
    {
        return true;
    }
  }
  return false;
}
```

כאשר:
1. p_1,p_2 הם פרדיקטים המכריעים את השפות L_1,L_2 בהתאמה.
2. length(s) היא פונקציה המחזירה את אורך המחרוזת s (מספר התווים במחרוזת).
3. subStr(s,st,en) היא פונקציה המחזירה את תת המחרוזת המתחילה בתו ה-st ומסתיימת בתו הen של המחרוזת s.

נשים לב, שהפונקציות length ו- subStr הן פונקציות חשיבות ומוגדרות היטב.

הוכחת נכונות:
<u>עצירה</u>: אורך המחרוזת c הוא סופי. לכן p תעבור על כל המחרוזות בזמן סופי ובסופו של דבר תיעצר (כשהתנאי שבתוך הלולאה יתקיים או כשהלולאה for תיעצר). בסיום הריצה תמיד יוחזר ערך T או F.

<u>הערך המוחזר נכון</u>: אם c הוא שרשור של 2 מחרוזות, אחת מ- L_1 והשנייה מ- L_2 , לאחר שהאינדקס i יגיע לאורך המחרוזת הראשונה בשרשור, חלוקת המחרוזות תהיה בדיוק ל-2 תת-המחרוזות המקוריות. לכן, עבור תת מחרוזת אחת, p_1 יחזיר T ועבור השנייה p_2 יחזיר T. מכאן, שהתנאי בתוך הלולאה יהיה גם T, הלולאה תפסק ו-p יחזיר T.
אם c אינו שרשור של 2 המחרוזות, בכל שלב של ריצת הלולאה, לפחות אחד הפרדיקטים p_1,p_2 יחזיר F על אחד מתתי-המחרוזות. לכן, התנאי בתוך הלולאה תמיד יהיה F, הלולאה תסתיים באופן טבעי ו-p יחזיר F.

d. **Kleene Star**

נראה שR סגור גם עבור Kleene Star, ע"י הגדרת פרדיקט שיכריע שרשור של 2 שפות. פעולת הפרדיקט: הפרדיקט מקבל מחרוזת s ומתחיל את הבדיקה. מאפסים אינדקס i להיות 0 ובודקים האם תת המחרוזת עד האינדקס i שייכת לשפה (באמצעות הפרדיקט המקורי של L). אם לא, מגדילים את i ב-1 וחוזרים על הבדיקה בשנית. אם כן, חותכים את תת-המחרוזת מתחילת s וממשיכים בסבב בדיקות חדש על המחרוזת שנשארה וכך הלאה. פעולת הבדיקה מתבצעת בדומה לרקורסיה. לצורכי נוחות נתייחס לכל "עומק" של רקורסיה כמעגלים, כך שהמעגל הראשון הוא הבדיקה המקורית. אם הגענו למצב בו באחד משלבי הרקורסיה, האינדקס המקומי הגיע לסוף המחרוזת והיא אינה שייכת ל-L, נחזור "מעגל" אחד אחורה, נקדם את האינדקס במעגל זה ב-1 ונמשיך בבדיקה. אם בשלב מסוים הגענו לסוף המחרוזת וגם זנב המחרוזת האחרון שייך ל-L, נחזיר T. אם נגיע לסוף המחרוזת במעגל החיצוני וגם המחרוזת השלמה s אינה שייכת לשפה (בבדיקה האחרונה האפשרית), נחזיר F.

הוכחת נכונות:

<u>עצירה</u>: אורך המחרוזת c הוא סופי. לכן p תעבור רקורסיבית על כל המחרוזות בזמן סופי ובסופו של דבר תיעצר (כאשר נגיע באחד המעגלים לסוף המחרוזת והיא תהיה שייכת לשפה או כאשר במעגל הראשון נגיע לסוף והיא תהיה שייכת/אינה שייכת לשפה).
בסיום הריצה תמיד יוחזר ערך T או F.

<u>הערך המוחזר נכון</u>: אם c הוא שרשור של מספר מחרוזות סופי מ-L, הרקורסיה תמצא חלוקה זו (כי אנו למעשה עוברים על כל החלוקות האפשריות למחרוזת). במקרה כזה, עבור כל תת מחרוזת בחלוקה, הפרדיקט של L יחזיר T ולכן, במעגל האחרון של אפשרות זו יוחזר גם לזנב המחרוזת T ולכן גם הבדיקה הכללית של s תחזיר T.
אם c אינו שרשור של מחרוזות מ-L, בכל אפשרות חלוקה תהיה לפחות תת מחרוזת אחת שאינה שייכת ל-L ועבורה יוחזר F. לכן, נגיע לאחר כל הבדיקות לבדיקה האחרונה, s כמחרוזת שלמה אינה שייכת ל-L ולכן גם בבדיקה זו יוחזר F וכפועל יוצא הפרדיקט שהגדרנו יחזיר גם F כנדרש.

5. קבע לכל אחת מהשפות האם היא שייכת ל-R, co-RE/R,RE/R או לאף אחת מהקבוצות. הוכח נכונות.

a. **קלט:** תוכנית p. **שאלה:** האם קיים x עבורו p עוצרת?

פתרון: נראה ש זוהי שפה הכריעה למחצה **(שייכת ל-RE/R)**.

i. **טענה:** זוהי שפה הכריעה למחצה **(שייכת ל-RE/R)**.

הוכחה:
נראה אלגוריתם המשתמש בשיטת השבלול באמצעות סטפר. יהי $s_1, s_2, \ldots$ קלטים אפשריים עבור p. נרוץ על הקלטים באופן הבא: עבור כל $i \in \mathbb{N}$ נרוץ על הקלטים $s_1, \ldots, s_i$ למשך i צעדים. במידה ועבור אחד הקלטים הסטפר עצר באופן טבעי, נחזיר T ונסיים את הבדיקה, אחרת נמשיך הלאה.

הוכחת נכונות:
במידה וקיים קלט x עבורו התוכנית p עוצרת, נגיע לקלט זה לאחר i_1 צעדים.
כיוון שהתוכנית p עוצרת עבור קלט זה, קיים i_2, מספר צעדים שעבורו p(x) עוצרת באופן טבעי. לכן, כאשר האלגוריתם שלנו יגיע ל$i=\max\{i_1, i_2\}$ הסטפר יעצור באופן טבעי והתוכנית תחזיר T כנדרש.
במידה ולא קיים קלט x עבורו p עוצרת, הסטפר לא ייעצר באופן טבעי בשום שלב ולכן התוכנית תתבדר.

ii. **טענה:** זוהי <u>אינה</u> שפה כריעה **(<u>אינה</u> שייכת ל-R)**.

הוכחה:
נניח בשלילה שהשפה כן כריעה ונראה רדוקציה לhalt. כלומר קיים פרדיקט stopsForSomeX המכריע את השפה (מחזיר T אם בשפה קיים x שעבורה התוכנית עוצרת ו-F אחרת).

Halt(p,x) := stopsForSomeX (g) where g:= λz.if z=x then return p(x)
else return loopy()

הוכחת נכונות:
<u>התוכנית עוצרת</u>: אנו מניחים כי stopsForSomeX כריעה ולכן תעצור עבור כל קלט. Halt מוגדרת להיות הערך ש-stopsForSomeX מחזירה ולכן תמיד תעצור (כי ע"פ ההנחה stopsForSomeX תמיד עוצרת).

<u>הערך המוחזר נכון</u>: אנו רוצים לבדוק האם התוכנית p עוצרת עבור x. התוכנית אינה עוצרת בוודאות עבור קלטים השונים מ-x. כאשר x ניתן כקלט ל-g, התוכנית תעצור רק אם p(x) עוצרת. כלומר, אם p(x) עוצרת, קיים x עבורו p עוצרת ולכן stopsForSomeX יחזיר T כנדרש. אחרת, g לא תעצור עבור אף קלט ו-stopsForSomeX תחזיר F.

b. **קלט:** תוכנית p. **שאלה:** האם כל מספר זוגי הגדול מ-2, ניתן להציג כסכום של 2 מספרים ראשוניים?

פתרון: נראה שזוהי שפה כריעה **(שייכת ל-R)**.

הנחת גולדבאך אומנם עדיין לא הוכחה או הופרכה אך באופן כללי לשאלה זו יש תשובה יחידה T או F. לכן שפה זו היא שפה טריוויאלית: אין תלות בקלט p ובהכרח ניתן להכריע אותה ע"י התוכנית $\lambda p,x.T$ או $\lambda p,x.F$. נשים לב, שאין צורך שנדע את התשובה לשפה טריוויאלית, כמו בדוגמה של שפת חברי הכנסת ה-18 שטרם הוכרעה רשמית (נכון לכתיבת שורות אלו) כפי שהוצגה בכיתה.

c. **קלט:** תוכנית p ו-2 משתנים: x ו-y. **שאלה:** האם p עוצרת בדיוק על 1 משני המשתנים?

פתרון: נראה ששפה זו אינה שייכת לאף אחת מהקבוצות R,RE/R,co-RE/R.

i. **טענה:** שפה זו <u>אינה</u> שייכת ל-R <u>ואינה</u> שייכת ל-co-RE/R.

הוכחה:
נוכיח באמצעות רדוקציית מיפוי. נניח בשלילה שקיים haltsOnOne(p,x,y) ונראה רדוקציה מ-halt.

Halt (p,x) := haltsOnOne(f(p,x)) where f:= λp,x.<g,x,ε>
g:= λz.if (z==x) then return p(x)
else return loopy()

הוכחת נכונות:
<u>התוכנית עוצרת</u>: אנו מניחים כי haltsOnOne כריעה ולכן תעצור עבור כל קלט. Halt מוגדרת להיות הערך שhaltsOnOne מחזירה ולכן תמיד תעצור (כי ע"פ ההנחה haltsOnOne תמיד עוצרת).

<u>הערך המוחזר נכון</u>: אנו רוצים לבדוק האם התוכנית p עוצרת עבור x. התוכנית g אינה עוצרת עבור כל ערך השונה מ-x. כמו כן, עבור קלט השווה ל-x, g מחזירה את p(x) ולכן תעצור רק במקרה ו-p(x) עוצרת. נשים לב: כאשר p(x) אינה עוצרת, g אינה עוצרת על 2 הקלטים ולכן haltsOnOne תחזיר F כנדרש. כאשר p(x) עוצרת, g תעצור רק עבור x ולכן haltsOnOne תחזיר T כנדרש.

הראנו כי קיימת רדוקצית מיפוי מ-halt ל-haltsOnOne זאת בסתירה לכך ש-halt אינה כריעה. לכן, גם haltsOnOne אינה כריעה. היות וhalt גם אינה שייכת ל- Co-RE/R נוכל להסיק בצורה דומה שגם haltsOnOne אינה שייכת ב-Co-RE/R.

ii. **טענה:** שפה זו <u>אינה</u> שייכת ל-RE/R.

הוכחה:
נוכיח באמצעות רדוקציית מיפוי.

$\overline{halt}$ (p,x) := haltsOnOne(f(p,x)) where f:= λp,x.<g,x,ε>
g:= λz.if (z==x) then return p(x)
else return T

הוכחת נכונות:
<u>התוכנית עוצרת</u>: אנו מניחים כי haltsOnOne כריעה ולכן תעצור עבור כל קלט.

$\overline{halt}$ מוגדרת להיות הערך שhaltOnOne מחזירה ולכן תמיד תעצור (כי ע"פ ההנחה haltsOnOne תמיד עוצרת).

<u>הערך המוחזר נכון</u>: אנו רוצים לבדוק האם התוכנית p אינה עוצרת עבור x. התוכנית g עוצרת עבור כל ערך השונה מ-x. כמו כן, עבור קלט השווה ל-x, g מחזירה את p(x) ולכן תעצור רק במקרה ו-p(x) עוצרת. נשים לב: כאשר p(x) אינה עוצרת, g עוצרת רק על קלט 1 ולכן haltsOnOne תחזיר T כנדרש. כאשר p(x) עוצרת, g תעצור על 2 הקלטים ולכן haltsOnOne תחזיר F כנדרש.

הראנו כי קיימת רדוקצית מיפוי מ- $\overline{halt}$ ל-haltsOnOne. היות ו $\overline{halt}$ אינה שייכת ל-RE/R, נסיק כי גם haltsOnOne אינה שייכת לקבוצה זו.

סה"כ הראנו ב-2 הטענות כי השפה אינה שייכת לאף קבוצה.

d. **קלט:** תוכנית p. **שאלה:** 3<|L(p)|?

פתרון: נראה שזוהי שפה הכריעה למחצה **(שייכת ל-RE/R)**.

i. **טענה:** זוהי שפה הכריעה למחצה **(שייכת ל-RE/R)**.

הוכחה:
נראה אלגוריתם המשתמש בשיטת השבלול באמצעות סטפר. יהי $s_1, s_2, \ldots$ קלטים אפשריים עבור p. נרוץ על הקלטים באופן הבא: עבור כל $i \in \mathbb{N}$ נרוץ על הקלטים $s_1, \ldots, s_i$ למשך i צעדים. בכל פעם שעבור אחד הקלטים הסטפר עצר באופן טבעי לראשונה, נקדם מונה ב-1. אם המונה יגיע ל-4 נחזיר T ונסיים את הבדיקה, אחרת נמשיך הלאה.

הוכחת נכונות:
במידה וקיימים יותר מ-3 קלטים עבורם p מחזירה T, נגיע לפחות ל-4 מהם לאחר i_1 צעדים. כיוון שהתוכנית p מחזירה T בוודאות עבור כל אחד מהקלטים הללו (ולכן בהכרח עוצרת), קיים i_2, מספר צעדים שעבורו p עוצרת עבור כל אחד מהקלטים באופן טבעי ומחזירה T. לכן, כאשר האלגוריתם שלנו יגיע ל$i=\max\{i_1, i_2\}$ הסטפר יעצור בפעם הרביעית, המונה יתעדכן ל-4, והתוכנית תחזיר T כנדרש.
במידה ולא קיימים לפחות 4 קלטים עבורם p מחזירה T, הסטפר לא ייעצר באופן טבעי מספיק פעמים והמונה לא יגיע ל-4. לכן, התוכנית תמשיך לרוץ, ובמילים אחרות - התוכנית תתבדר.

ii. **טענה:** זוהי <u>אינה</u> שפה כריעה **(<u>אינה</u> שייכת ל-R)**.

הוכחה:
נניח בשלילה שהשפה כן כריעה ונראה רדוקציה מhaltn. כלומר קיים פרדיקט moreThan3 המכריע את השפה (מחזיר T אם בשפה יש יותר מ-3 קלטים שעבורם התוכנית p עוצרת, ו-F אחרת).

```
Halt(p,x) := moreThan3(g) where g:= λz.if z=x then return p(x)
                                   else if z=next(x) then T
                                   else if z=next(next(x)) then T
                                   else if z=next(next(next(x))) then T
                                   else return loopy()
```

הוכחת נכונות:
<u>התוכנית עוצרת</u>: אנו מניחים כי moreThan3 כריעה ולכן תעצור עבור כל קלט. Halt מוגדרת להיות הערך שmoreThan3 מחזירה ולכן תמיד תעצור (כי ע"פ ההנחה moreThan3 תמיד עוצרת).

<u>הערך המוחזר נכון</u>: אנו רוצים לבדוק האם התוכנית p עוצרת עבור x. התוכנית g עוצרת בוודאות עבור 3 קלטים. עבור יתר הקלטים (פרט לקלט x), תכנס g ללולאה אינסופית. כאשר נשלח ל-g הקלט x מוחזר הערך של p(x). לכן, אם עבור x התוכנית p עוצרת, הרי ש-g תעצור עבור יותר מ-3 קלטים ולכן יוחזר T כנדרש. אחרת, p תעצור עבור 3 קלטים בדיוק ולכן יוחזר F כנדרש.

e. **קלט:** תוכנית p. **שאלה:** $|L(p)| \le 3$?

פתרון: זוהי שפה שהמשלים שלה היא שפה הכריעה למחצה **(שייכת ל-Co-RE/R)**.

i. **טענה:** זוהי שפה שהמשלים שלה היא שפה הכריעה למחצה **(שייכת ל-Co-RE)**

הוכחה: נשים לב כי המשלים הוא $|L(p)|>3$. כפי שהוכחנו בסעיף הקודם, המשלים שייך ל-RE/R ואינו שייך ל-R. לכן, באופן מיידי (ע"פ מה שהוכחנו בכיתה) נסיק כי שפה זו שייכת ל-Co-RE.

טענה: זוהי <u>אינה</u> שפה כריעה **(<u>אינה</u> שייכת ל-R)**.

הוכחה:
נניח בשלילה שהשפה כן כריעה. מכאן שגם המשלים של השפה שייך ל-R אך בסעיף הקודם הראנו כי הוא אינו שייך ל-R. קיבלנו סתירה ולכן השפה אינה שייכת ל-R.

f. **קלט:** פרדיקט P. **שאלה:** האם L(P) היא כריעה?

פתרון: נראה ששפה זו אינה שייכת לאף אחת מהקבוצות R,RE/R,co-RE/R.

i. **טענה:** שפה זו <u>אינה</u> כריעה ואינה שייכת ל-co-RE/R.

הוכחה:
נוכיח באמצעות רדוקציית מיפוי מ-halt:
נניח שהשפה כריעה ומוכרעת ע"י הפרדיקט Q. נסמן את הפרדיקט של השפה $halt\varepsilon$ כ-$P\varepsilon$ (שפת התוכניות שעוצרות עבור הערך אפסילון).

$halt$ (p,x) := Q(g(p,x)) where

$$g := \lambda p,x.\lambda z.P\varepsilon(z) \vee f(z)$$
$$f := \lambda z.\begin{cases} T & p(x)=p(x) \\ loopy() & o.w. \end{cases}$$

הוכחת נכונות:
<u>התוכנית עוצרת</u>: אנו מניחים כי Q כריעה ולכן תעצור עבור כל קלט. התוכנית g תעצור בוודאות כיוון שהיא בסך הכל מחזירה תוכנית חדשה (ולא מפעילה תוכנית זו עדיין). $halt$ מוגדרת להיות הערך שQ מחזירה ולכן תמיד תעצור (כי ע"פ ההנחה Q תמיד עוצרת).

<u>הערך המוחזר נכון</u>: אנו רוצים לבדוק האם התוכנית p אינה עוצרת עבור x. אם p עוצרת על x, ל-Q יועבר הפרדיקט $\lambda z.P\varepsilon(z) \vee f(z) = \lambda z.P\varepsilon(z) \vee T = T$, כלומר הפרדיקט המחזיר T לכל ערך. במקרה כזה, השפה המוגדרת ע"י הפרדיקט היא שפת כל המחרוזות והיא כמובן שפה כריעה. לכן Q יחזיר T כנדרש. אם p אינה עוצרת על x, ל-Q יועבר הפרדיקט $\lambda z.P\varepsilon(z) \vee f(z)$.
בעת בדיקת ערך בפרדיקט זה, אם הערך שייך ל- $halt\varepsilon$, יוחזר T (והתנאי השני לא ייבדק). אם הערך לא שייך ל- $halt\varepsilon$, יוחזר מהפרדיקט של $halt\varepsilon$ F ולכן ננסה לבדוק באמצעות f(z). במקרה זה, כיוון ש-p אינה עוצרת, הפרדיקט כולו יתבדר (**הערה:** כדי לא לסבך את הקוד השתמשתי בתנאי "או" פשוט. כיוון ש- $halt\varepsilon$ אינה כריעה, נאלץ להשתמש בסטפר עד שאחת הפונקציות תחזיר T). כלומר, עבור קלטים השייכים ל- $halt\varepsilon$ יוחזר T ועבור קלטים שאינם שייכים ל- $halt\varepsilon$ הפרדיקט יתבדר. לכן, השפה שתוגדר ע"י הפרדיקט היא בעצם $halt\varepsilon$. זו שפה שאינה כריעה ולכן Q יחזיר F כנדרש.

הראנו כי קיימת רדוקצית מיפוי מ- $halt$ ל-Q. היות ו $halt$ אינה כריעה ואינה שייכת ל-co-RE/R, נסיק כי גם Q אינה שייכת לקבוצה זו.

ii. **טענה:** שפה זו <u>אינה</u> שייכת ל-RE/R.

הוכחה:
נסמן את השפה כ L_R . בכיתה הראנו והוכחנו כי קיימת רדוקצית מיפוי מ-Halt לשפה המשלימה $L_{\overline{R}}$. כלומר מתקיים $halt \le_m L_{\overline{R}}$ ומכאן (ע"פ מה שהוכחנו בכיתה) גם מתקיים $\overline{halt} \le_m \overline{L_{\overline{R}}} \Rightarrow \overline{halt} \le_m L_R$.

היות ו- $\overline{halt}$ אינה כריעה ואינה שייכת ל-RE/R נסיק כי גם L_R אינה שייכת כריעה ואינה שייכת ל-RE/R.

הראנו כי קיימת רדוקצית מיפוי מ-halt ל-haltsOnOne זאת בסתירה לכך ש-halt אינה כריעה. לכן, גם haltsOnOne אינה כריעה. היות וhalt גם אינה שייכת ל-Co-RE/R נוכל להסיק בצורה דומה שגם haltsOnOne אינה שייכת ב-Co-RE/R.

סה"כ הראנו ב-2 הטענות כי השפה אינה שייכת לאף קבוצה.

g. **קלט:** פרדיקט p. **שאלה:** האם L(P) מוכרעת למחצה?

פתרון: זוהי שפה כריעה **(שייכת ל-R)**.

הוכחה:
ע"פ ההגדרה, כל שפה L(p) מוכרעת למחצה ע"י הפרדיקט שלה p. כלומר, השפה L(P) מוגדרת להיות כל הערכים להם P מחזיר אמת. לכן, השפה L(p) מעצם ההגדרה מוכרעת למחצה ע"י P. מכאן ששאלה זו היא שאלה טריוויאלית ומוכרעת ע"י הפרדיקט $\lambda x.T$. הראנו כי קיים פרדיקט המכריע את השפה ולכן שפה זו שייכת ל-R.

h. **קלט:** תוכנית p ללא קלט, שמקיימת |p|<bb(1000).
שאלה: האם p עוצרת?

פתרון: זוהי שפה כריעה **(שייכת ל-R)**.

הוכחה:

מספר התוכניות ללא קלט שמקיימות את התנאי |p|<bb(1000) הוא מספר סופי. לכן, קיימים מספר סופי של שאילתות שניתן לבצע על הבעיה. ע"פ מה שנלמד בכיתה, כל בעיה עם מספר סופי של שאילתות היא בעיה כריעה (בדומה לבעיית חברי הכנסת ה-18). לכן, זוהי שפה כריעה.

הערה: נשים לב, שמשפט רייס אינו תופס במקרה זה כיוון שהבעיה אינה סמנטית. תהי f תוכנית עוצרת באורך bb(1000). התוכנית $g = \lambda x.f(x)$ שקולה לתוכנית f היות והערכים שהיא מחזירה זהים לערכים של f. עם זאת, ברור שהאורך של g גדול מbb(1000). לכן, למרות שהתוכניות שקולות סמנטית, הבעיה תחזיר עבור g F כי היא אינה שייכת לשפה.

6. **נתון:** $L_1, L_2 \in RE/R$. **הוכח** האם הביטויים הבאים אפשריים:

a. $L_1 \cup L_2 \in R$

פתרון: מצב זה הוא מצב אפשרי.

הוכחה:
נגדיר

$$L_1 = \{<p,x> \mid p \quad halts \quad on \quad x \quad or \quad |p| \quad is \quad even\}$$
$$L_2 = \{<p,x> \mid p \quad halts \quad on \quad x \quad or \quad |p| \quad is \quad odd\}$$

L_1 ו-L_2 הן שפות כריעות למחצה (כלומר $L_1, L_2 \in RE/R$): פונקצית האורך היא פונקציה מוגדרת היטב שתמיד עוצרת. כמו כן הפרדיקטים odd ו-even הם פרדיקטים שעוצרים תמיד ומוגדרים היטב. עם זאת הראנו ש-halt היא בעיה הכריעה למחצה (כאשר התוכנית עוצרת נוכל תמיד בקלות להחזיר T אך כאשר התוכנית אינה עוצרת, גם הפרדיקט המכריע את halt יתבדר).
נשים לב כי $L_1 \cup L_2 = \{<p, x>\}$. כלומר, זוהי שפה טריוויאלית של כל התוכניות והקלטים ולכן מוכרעת על ידי הפרדיקט $\lambda p, x.T$. פרדיקט זה תמיד עוצר ומחזיר T לכל תוכנית ולכן מתקיים $L_1 \cup L_2 \in R$ כנדרש.

b. $L_1 \cup L_2 \in R$ וגם $L_1 \cap L_2 \in R$

פתרון: מצב זה אינו אפשרי.

הוכחה:
נניח בשלילה שהמצב כן אפשרי. היות ו-$L_1 \cup L_2 \in R$ קיים פרדיקט p_1 המכריע את השפה $L_1 \cup L_2$. כמו כן, היות ו-$L_1 \cap L_2 \in R$ קיים פרדיקט p_2 המכריע את השפה $L_1 \cap L_2$. נראה כעת כי ניתן להכריע את השפה L_1 באמצעות הגדרת פרדיקט:

$$P_{l1}(x) = \begin{cases} T & p_1(x) \wedge p_2(x) \\ F & \neg p_1(x) \\ g(x) & p_1(x) \wedge \neg p_2(x) \end{cases}$$

כאשר g(x) היא תוכנית המקבלת מחרוזת x ומשתמשת בסטפר בצורה הבאה: בכל פעם היא מפעילה את הסטפר i פעמים לחילופין על הפרדיקט שמכריע את L_1 והפרדיקט שמכריע את L_2. אם הפרדיקט של L_1 עוצר באופן טבעי בשלב מסויים ומחזיר T, g מחזירה T. אם L_2 עוצר באופן טבעי בשלב מסויים ומחזיר T, g מחזירה F.

הוכחת נכונות:
<u>התוכנית עוצרת</u>: שפות האיחוד והחיתוך הן שפות כריעות על פי ההנחה. לכן, הפרדיקטים המכריעים אותן יעצרו בוודאות על כל ערך. כמו כן, הפונקציה g תעצור תמיד כיוון שאם x שייך לאיחוד ואינו שייך לחיתוך, הוא שייך רק לאחת מהשפות. לכן, בשלב מסוים הפרדיקט של השפה אליה הוא שייך יסתיים באופן טבעי ויחזיר T (L1 ו-L2 הן שפות כריעות למחצה לכן אחד הפרדיקטים חייב להסתיים).

<u>התשובה המוחזרת נכונה</u>: אם x שייך לאיחוד ולחיתוך, הוא שייך ל-2 השפות בוודאות ובפרט שייך ל-L1. לכן, P מחזירה T כנדרש. אם x לא שייך לאיחוד, x אינו שייך לאף אחת מהשפות ובפרט אינו שייך ל-L1, לכן, p מחזירה F כנדרש. אם x שייך לאיחוד אך לא שייך לחיתוך, אז x שייך בדיוק לאחת השפות. התוכנית g קובעת לאיזה מהשפות הוא שייך ע"י הרצת סטפר. כיוון שL1 ו-L2 הן שפות כריעות למחצה, הפרדיקט המכריע את השפה אליה שייך x בטוח יעצור ויחזיר T (והוא היחיד שיחזיר T עבור x מהשתיים). לכן, אם x שייך ל-L1, הפרדיקט של L1 בטוח יעצור ויחזיר T ולכן גם g תחזיר T כנדרש ואם x אינו שייך ל-L1, הפרדיקט של L2 בטוח יעצור ויחזיר T ולכן g תחזיר F כנדרש.

סה"כ הראנו שניתן להכריע את L1 בסתירה לנתון ש $L_1 \in RE/R$. לכן, ההנחה בשלילה אינה מתקיימת והמצב אינו אפשרי.