שפות תכנות, 236319

פרופ' ד. לורנץ
אביב 2025



תרגיל בית 2

תאריך פרסום: 22.4.25

מועד אחרון להגשה: 8.5.25 בשעה 23:59.

מתרגל אחראי: ענבר מנצ'ל

מייל: inbarmenchel@campus.technion.ac.il

הערות:

- שאלות על התרגיל יענו בפורום הקורס בפיאצה **בלבד**.
- ניתן לפנות במייל במקרה של בעיות אדמיניסטרטיביות פרטניות, על כותרת המייל להתחיל ב-"*236319 - תרגיל בית 2*"
- הבהרות ותיקונים לתרגיל יפורסמו בגיליון זה ויסומנו בצהוב.
- לא תתאפשר הגשה לאחר מועד ההגשה כלל (אין הגשות באיחור)
- הסיבה היחידה שבגינה ניתן לקבל דחייה במועד הגשת תרגילי הבית היא מילואים. כדי לקבל דחיה עבור שירות מילואים, יש לפנות למתרגל האחראי על התרגיל במייל.
- הגשה בזוגות. לפני ההגשה, ודאו כי ההגשה שלכם תואמת את הנחיות ההגשה בסוף התרגיל. **הגשות שלא יעמדו בהנחיות יפסלו על הסף**.

חלק יבש

לפניכם מספר שאלות, חלקן על דברים שראינו בכיתה וחלקן לא. על מנת לענות על השאלות ניתן להיעזר בחיפוש באינטרנט **ובבינה מלאכותית** ✨. כתבו את הפתרונות אליהן בקובץ **dry.pdf**.

שאלה 1

לאחר שארה"ב הטילה מכסים על הייבוא, הבורסות בעולם חוו ירידות ערך חריגות, וכתוצאה מכך, מערכות המחשוב שלהן קרסו (דבר דומה קרה גם ב-2008, כפי שאתם יכולים לראות כאן). לצערם של הסוחרים, מערכות המחשוב בבורסה הן ישנות מאוד וכתובות בשפה עתיקה בשם-Tariff Lisp. ולכן, הם פנו אליכם, מתכנתים בכירים בבורסה, כדי שיוכלו לעזור להם לשקם את המערכת.

לפניכם דקדוק של Tariff Lisp הנתון כ-EBNF:

```
<program> ::= <s_expression>

<s_expression> ::= <atomic_symbol>
                 | "(" <s_expression> "." <s_expression> ")"
                 | <list>

<list> ::= "(" [ <s_expression> ] { " " <s_expression> } ")"

<atomic_symbol> ::= <letter> <atom_part>
<atom_part> ::= <letter> <atom_part>
                  | <number> <atom_part>
                  | ""
<letter> ::= [a-z]
<number> ::= [1-9]
```

1. רשמו את רשימת הטרמינלים, הלא טרמינלים וסימבול התחלתי של הדקדוק.
2. עבור כל אחת מסדרות הטרמינלים הבאות קבעו האם היא שייכת לשפה המוגדרת ע"י הדקדוק. אם כן, פרטו את כללי הגזירה המובילים אליה. אם לא, הסברו בקצרה מדוע.

```
1.()
2.(cons (a b))
3.(cons 1 (b c))
4.(a b) c)
5.(car (cons giveyou nevergonnaup))
6.((define m k) (cons 1 (t c)) . (car cdr))
```

דוגמה:

`(a.b)` - שייך לדקדוק. גזירה חוקית:

```
<s_expression>
    ⇒ "(" <s_expression> "." <s_expression> ")"
    ⇒ "(" <atomic_symbol> "." <atomic_symbol> ")"
    ⇒ "(a.b)”
```

3. האם הדקדוק מכיל ambiguities? אם כן, רשמו סדרת טרמינלים והדגימו את ה-ambiguity. אם לא, הסבירו מדוע. ✨

שאלה 2 ✨

1. עבור כל אחת מהקריאות הבאות ציינו את הודעת השגיאה שתתקבל ב-SML והסבירו בקצרה את מהותה.
2. ציינו באילו מהקריאות ChatGPT טועה כאשר שואלים אותו מה התשובה ונסו לשער מדוע הוא טועה.

```
1.  "Rick" + "Astley";
2.  13 / 4;
3.  fun f x = if x = 0 then x else 1.0;
4.  fun f x = if x = #"r" then x ^ "t" else x ^
    "k";
5.  fun super_secret_idea(n, p, t) =
        if n=1 then (if t=1.2 then p else 1+p) else
        super_secret_idea (n-1, n*p,t);
6.  5 + (-3)-2;
7.  infix 10 lol;
8.  fun sqrt_of_int z:int = Math.sqrt (real z);
9.  Math.sqrt 69;
10. val infixr = 35;
```

חלק רטוב

- קבצים המסופקים לחלק זה תוכלו למצוא בגיטהאב של הקורס Homework2-
- פתרון לכל אחת מהשאלות יש לכתוב בקובץ נפרד ששמו hw2_q<i>.sml עבור שאלה i. בנוסף לכל שאלה מסופקת לכם דוגמת קלט ופלט רצוי. לצורך העניין את הפתרון שלכם לשאלה 1 תוכלו לבדוק בעזרת הפקודות הבאות:

```
smlnj hw2_q1.sml < q1.in | sed '1,/===TEST START===/d' >
q1.out
diff q1.out q1.expected
```

- סיפקנו עבורכם סביבת פיתוח מוכנה לSML מוכנה ב-VSCode. מדריך התקנה ושימוש.
- יש לבדוק, לקמפל ולהריץ את פתרונותיכם לתרגיל זה בסביבה הנ"ל.

שאלה 1 ✨

כתבו פונקציות בשמות sign,...,sig2,sig1 כאשר הפונקציה sigi מתאימה לסעיף i. למשל, הפונקציה sig4 תהיה תשובתך לאתגר בסעיף 4.

הבהרה - **אסור** להשתמש ב-pattern matching ו-type constraints (כלומר דרישה בחתימה על טיפוס החזרה או טיפוס הפרמטר של הפונקציה).

אין חשיבות למה שהפונקציה מבצעת. כל שעליכם להבטיח הוא כי חתימת הפונקציה תהיה כנדרש. אך לצרכי תרגול, מומלץ לחשוב על פונקציות בעלות משמעות.

```
1.  ('a * 'b -> 'c) -> 'a -> 'b -> 'c -> ('d ->
  bool) -> 'd -> 'c
2.  real -> real -> bool
3.  ('a -> 'b) -> 'a list list -> 'b list list
4.  unit -> int
5.  ('a -> 'b) -> ('b -> 'c) -> 'a -> 'd -> 'c
6.  ('a -> 'b) -> 'a list -> 'c -> 'b list
7.  bool -> int -> 'a -> int -> 'a
8.  ('a -> 'b) -> ('a -> 'b -> 'c) -> 'a -> 'c
9.  ('a -> 'b) -> ('b -> 'c) -> 'a -> 'c
10. ('a -> bool) -> (('a -> bool) -> 'b) -> 'a ->
  'b -> 'b
```

שאלה 2

ממשו את הפונקציות:

```
curry: ('a * 'b -> 'c) -> 'a -> 'b -> 'c
uncurry: ('a -> 'b -> 'c) -> 'a * 'b -> 'c
```

כפי שראינו בתרגולים, כל פונקציה ב-SML מקבלת ארגומנט יחיד ומחזירה ערך יחיד. למרות זאת, יש ב-SML שתי דרכים לדמות פונקציה המקבלת שני פרמטרים:

- Uncurried - פונקציה המקבלת tuple של שני איברים ומבצעת את הפעולה הנדרשת (ניתן להגיד שבכל זאת הפונקציה מקבלת פרמטר אחד מסוג tuple).
- Curried - פונקציה המקבלת את הארגומנט הראשון ומחזירה פונקציה שניה המקבלת את הארגומנט השני ומבצעת את הפעולה הנדרשת על שני הארגומנטים.

לכל אחת מהאפשרויות יש יתרונות וחסרונות משלה וניתן לבחור באחת האפשרויות ע"פ הצורך. כמו כן, ניתן להמיר פונקציה שהיא uncurried להיות curried ולהפך. בשאלה זו עליכם להגדיר שתי פונקציות המבצעות המרה מהאפשרות הראשונה לשנייה ולהפך.

הפונקציה curry ממירה פונקציה מ-uncurried ל-curried.

הפונקציה uncurry ממירה פונקציה מ-curried ל-uncurried.

דוגמאות הרצה:

```
- curry op*;
val it = fn : int -> int -> int
- val a = it 3 4;
val a = 12 : int
- uncurry it;
val it = fn : int * int -> int
- val a = it (3,4);
val a = 12 : int
```

שאלה 3

לאחר מסעות רבים באוברוורלד, החליט סטיב לנוח בביתו ולכתוב אודות מסעותיו בספר. לצערו של סטיב, הוא אינו יודע לכתוב, ומכיר רק מילים מועטות. בשאלה זו נעזור לסטיב לכתוב על מסעותיו בשפה שהוא מכיר.

סטיב יכול לכתוב אך ורק במשפטים המורכבים מצירופים מוכרים. הצירופים המוכרים הם צמדים של תחילית וסיומת אותם אמר סטיב במהלך המסע, ולכן הוא יכול לכתוב אותם. סטיב יכול לכתוב אך ורק תחיליות וסיומות אשר מתאימות זו לזו (כפי שיוגדר בהמשך).

הצירופים אותם סטיב מכיר הם (כאשר המבנה הוא start-end):

first we mine-then we craft
I-am Steve
as a child-I yearned for the mines
chicken-jockey
ender-pearl
flint-and steel
slime-cube
water bucket-release

כדי לעזור לסטיב לכתוב, נתנה לו חברתו אלכס שני כלי עזר:

1) *: הכוכבית מאפשרת לסטיב לא לכתוב צירוף. כלומר, ניתן לפרש אותה כצירוף כלשהוא.
2) totem of undying: כלי זה מאפשר לסטיב להשתמש בו במקום כל תחילית של כל צירוף שיבחר.

כעת, רוצה סטיב לבדוק האם הסיפור אותו כתב הוא חוקי. כאשר סיפור חוקי הוא אוסף של צירופים חוקיים או שימוש בעזרים אשר נתנה לו אלכס.

לשם הפתרון, העתיקו את הקוד הבא:

```
datatype start = first_we_mine | I | as_a_child | chicken |
ender | flint | slime | water_bucket;
datatype ending = jockey | pearl | and_steel |
I_yeared_for_the_mines | am_Steve | release | then_we_craft |
cube;
datatype part = part of (start * ending);
datatype content = Content_End  | Content of (part * content);
```

כתבו את הפונקציה הבאה:

```
good_part = fn : content -> string list -> bool
```

הפונקציה מקבלת:

1) את הסיפור שסטיב כתב.
2) רשימה של מחרוזות אשר מייצגות את הסיפור.

הפונקציה מחזירה האם אכן המחרוזת יכולה לייצג את הסיפור אשר סטיב כתב.

<u>הערות</u>:

- עבור סיפור שאינו חוקי, יש להחזיר false.
- לא ניתן להניח כי הרשימה אינה ריקה.

לדוגמא:

עבור הקלט הבא:

```
good_part
Content (part(first_we_mine, then_we_craft), Content_End)
["first we mine", "then we craft"];
```

יוחזר **true**, שכן הסיפורים מתאימים.

אבל, עבור:

```
good_part
Content (part(first_we_mine, and steel), Content_End)
["first we mine", "then we craft"];
```

יוחזר **false**, שכן הסיפורים לא מתאימים.

טבלה להתאמה בין ביטוי למחרוזת:

content	**string**
first_we_mine	first we mine
I	I
as_a_child	as a child
chicken	chicken
ender	ender
flint	flint
slime	slime

water_bucket	water bucket
jockey	jockey
pearl	pearl
and_steel	and steel
I_yeared_for_the_mines	I yearned for the mines
am_Steve	am Steve
release	release
then_we_craft	then we craft
cube	cube

הנחיות הגשה

- את הפתרון לתרגיל הבית יש להגיש בקובץ zip המכיל את הקבצים הבאים בלבד:

hw2_q1.sml, hw2_q2.sml, hw2_q3.sml, dry.pdf

- על החלק היבש להיות מוקלד, אין להגיש סריקה או צילום של התשובות לחלק זה.
- שם קובץ ההגשה יהיה EX2_ID1_ID2.zip כאשר ID1, ID2 הם מספרי ת.ז. של המגישים.
- בודקי התרגילים **מאוד** אוהבים Memes. שתפו את תחושותיכם במהלך פתירת התרגיל באמצעות Memes מתאימים ב-pdf של החלק היבש. Memes מצחיקים בצורה יוצאת דופן יזכו את היוצרים בבונוס.

בהצלחה!