שפות תכנות, 236319.

פרופ' י. גיל

חורף 2020-2021

תרגיל בית 1

תאריך פרסום: 27/10/2020
מועד אחרון להגשה: 10/11/2020
מועד אחרון להגשה מאוחרת: 14/11/2020
מתרגל אחראי: גיא שפירא
אי-מייל: guy.shapira@campus.technion.ac.il
בפניה בדוא"ל, נושא ההודעה (subject) יהיה "PL-EX1" (ללא המירכאות).

תרגיל בית זה מורכב משני חלקים, חלק יבש וחלק רטוב.
לפני ההגשה, **ודאו שההגשה שלכם תואמת את הנחיות ההגשה בסוף התרגיל.**
תיקונים והבהרות יפורסמו בסוף מסמך זה, אנא הקפידו להתעדכן לעתים תכופות.

חלק יבש

בכל השאלות מותר להשתמש בפונקציות האטומיות והסיפריה של מיני-ליספ, ובהן בלבד. (16 פונקציות בסה"כ).
מותר גם להשתמש בפונקציות שנראו בכיתה עם ציון שנלקחו משם.
מותר להשתמש בפונקציות עזר בתרגילים שאתם פותרים.

קובץ המכיל את הפונקציות שהוגדרו בתרגול badd bnormalize ופונקציות העזר שלהן נוסף לאתר הקורס תחת תרגיל בית 1, שם הקובץ הוא badd.txt. ניתן להשתמש בפונקציות הללו אך אין צורך להגיש אותן.

1. ממש את הפונקציה cat המקבלת שתי רשימות והמחזירה את השרשור של שתיהן.
2. פתור את תרגילים 1 ו-2 בספר הקורס[1]. ב COMMON LISP ישנה הפונקציה member. הסבר את ההבדל בינה ובין exists שמופיעה בספר הקורס, וממש את הפונקציה member.
3. פתור את תרגיל 3 בספר הקורס.
4. ממש את הפונקציה bmul הפועלת בדומה ל badd שנלמדה בתרגול, אלא שהיא מכפילה את שני המספרים הנתונים.
5. תמצת את אלגוריתם השיערוך של מיני-ליספ ותרגם אותו ל PSEUDO CODE באנגלית. האלגוריתם מופיע במספר תיבות בעברית בפרק על מיני-ליספ בספר הקורס.

חלק רטוב

מבוא ל-ML

הבהרה בעקבות מיילים מרובים- בחלק זה מותר לכתוב אך ורק את 8 הפונקציות שצויינו במשימה

את פתרון חלק זה שמרו בקובץ hw1.sml.

משימה 1 - חתימות ב-ML

כתוב פונקציות בשמות `sig1`, `sig2`, …, `sig`n כאשר הפונקציה `sig`i מתאימה לסעיף i. למשל, הפונקציה `sig4` תהיה תשובתך לאתגר בסעיף 4.

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

אין חשיבות למה שהפונקציה מבצעת. כל שעליך להבטיח הוא כי חתימת הפונקציה תהיה כבסעיף:

[1] מדובר בסיכום על מיני ליספ שנמצא בתיקייה ההרצאות באתר הקורס

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

משימה 2 – Curried & Uncurried

כל פונקציה ב-ML מקבלת ארגומנט יחיד ומחזירה ערך יחיד. אבל יש ב-ML שתי דרכים לדמות פונקציה המקבלת שני פרמטרים:

- Uncurried - פונקציה המקבלת tuple של שני איברים ומבצעת את הפעולה הנדרשת.
- Curried - פונקציה המקבלת את הארגומנט הראשון ומחזירה פונקציה 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 בקלות להמיר פונקציה שהיא uncurried להיות curried ולהפך. בשאלה זו עליכם להגדיר שתי פונקציות המבצעות המרה מהאפשרות הראשונה לשנייה ולהפך.

א. הפונקציה curry מקבלת פונקציה מהסוג הראשון (uncurried) וממירה אותה לפונקציה מהסוג השני (curried).

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

ב. הפונקציה uncurry המבצעת את הפעולה ההפוכה.

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

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
```

MLISP

הבהרה- אין להשתמש בפעולות מתוך ספריית List

הבהרה נוספת- בחלק זה ניתן להשתמש בפונקציות עזר בצורה חופשית

כללי

בתרגיל זה נממש בשלבים גרסה מצומצמת של המפרש של שפת LISP בשפת ML.

שלב א'

בתרגיל זה נממש מספר פונקציות עזר שיעזרו לנו במימוש המפרש. את הפתרון לתרגיל זה עליכם לכתוב בקובץ mlisp.sml.

סעיף 0

קראו את התיעוד של STRING וCHAR . יתכן ושימוש בפונקציות אלו יעזרו לכם לממש את המפרש.

מותר להשתמש בכל הפונקציות המופיעות בעמודים אלו במימוש תרגיל זה.

סעיף 1

ממשו את הפונקציה:

```
val isNumber = fn : string -> bool
```

הפונקציה הזו מקבלת מחרוזת ומחזירה האם המחרוזת מייצגת מספר. המחרוזת מייצגת מספר אם כל תו במחרוזת הוא ספרה.

דוגמאות הרצה:

```
- isNumber "700";

val it = true : bool

- isNumber "19500";

val it = true : bool

- isNumber "19z500";

val it = false : bool
```

```
- isNumber "00500";

val it = true : bool

- isNumber "";

val it = true : bool
```

סעיף 2

ממשו את הפונקציה:

```
val atoi = fn : string -> int
```

הפונקציה מקבלת מחרוזת שמובטח שהיא מייצגת מספר, ומחזירה את המספר שהיא מייצגת. (בדומה לפונקציה atoi שאתם מכירים משפות אחרות).

הערה:

שימו לב שלא נדרשת תמיכה במספרים שליליים. לשם פשטות, המתכנת שכותב קוד בשפת MLISP לא יוכל לציין מספרים שליליים באופן מפורש למשל ע"י 7- אבל תוצאות חישוב של המפרש יוכלו להניב מספרים שליליים כמו למשל הקוד:

(- 6 7)

שגורם למפרש לפלוט את הערך 1-. אין צורך לדאוג לכך עכשיו כמובן, בתרגילי ההמשך אתם תממשו זאת.

דוגמאות הרצה:

```
- atoi "700";

val it = 700 : int

- atoi "19500";

val it = 19500 : int

- atoi "00500";

val it = 500 : int

- atoi "";

val it = 0 : int
```

סעיף 3

אחד מהשלבים הראשונים בפעולת הקומפילייר או המפרש הוא פירוק הקוד לtokens. הtoken הוא יחידה בעלת משמעות תחבירית עבור המפרש. מכיוון שזהו אינו הקורס בקומפילציה אנחנו לא נתעמק בחלק זה וכל מה שיעניין אותנו הוא ליצור מפיסת קוד של MLISP רשימה (list) של tokens.

ממשו את הפונקציה:

```
val tokenize = fn : string -> string list
```

פונקציה זו מקבלת פיסת קוד בMLISP ומפרידה אותו לרשימה של tokens.

סוגי הtokens:

- ")" - סוגריים פותחים.
- "(" - סוגריים סוגרים.
- כל שאר המחרוזות נחשבות גם כן tokens כאשר ההפרדה ביניהם היא ע"י התו רווח (" ").

דוגמאות ההרצה יסייעו לכם להבין את הדרישה טוב יותר.

הערה:

הtokenizer המוגבל שלנו הוא למעשה החלק היחיד בשלב ה lexical analysis של המפרש שלנו (עוד על זה בקורס בקומפילציה). מכיוון שהוא מאוד פשוט, MLISP תתמוך במספרים בלבד ולא תממש סוכר סינטקטי מכל סוג שהוא. למשל, לא נוכל להגדיר ליטרלים המייצגים מחרוזת ולא נתמוך בQUOTE.

דוגמאות הרצה:

```
- val program = "(+ 2 3)";

val program = "(+ 2 3)" : string

- val tokens = tokenize program;

val tokens = ["(","+","2","3",")"] : string list

- val program = "(car (cons (+ 3 10) (+ 3 2)))";

val program = "(car (cons (+ 3 10) (+ 3 2)))" : string

- (tokenize program) =
["(","car","(","cons","(","+","3","10",")","(","+","3","2",")",")"
,")"];

val it = true : bool
```

הנחיות

- בתרגיל זה ניתן להשתמש רק בחומר שנלמד בשפת ML עד (וכולל) תרגול 2. אין להשתמש באף פונקציה או תכונה של השפה שלא נלמדה בתרגולים.
- רשימת הקבצים שצריכים להופיע בתוך קובץ ה-zip היא:
  `dry.pdf,hw1.sml,mlisp.sml`
- בכל קובץ קוד, הוסיפו בשורה הראשונה הערה המכילה את השם, מספר ת״ז וכתובת הדואר האלקטרוני של המגישים מופרדים באמצעות רווח.
- על החלק היבש להיות מוקלד, אין להגיש סריקה או צילום של התשובות לחלק זה.
- שם קובץ ההגשה יהיה EX1_ID1_ID2.zip כאשר ID1, ID2 הם מספרי ת.ז. של המגישים בסדר עולה ממש. אם לא ניתן לסדר את מספרי ת״ז בסדר שכזה יש לפנות למשרד הפנים.
- אין צורך להגיש ניירת הסמסטר. תא הקורס לא יבדק במהלך הסמסטר, אז אנא חסכו בנייר.
- בודקי התרגילים מאוד אוהבים Memes. שתפו את תחושותיכם במהלך פתירת התרגיל באמצעות Meme מתאים על דף השער בהגשה - אולי יצא מזה משהו מעניין!

בהצלחה!

תיקונים והבהרות

28.10.2020- שינוי בסוגריים בsig8,sig7 בשאלה 1.

1.11.20- הוספה דוגמה להפעלה של atoi במקרה קצה של מחרוזת ריקה + הערה על מיני ליספ.

3.11.20- הודעה על ספריית List.

6.11.20- הבהרות לגבי שימוש בפונקציות עזר.

08.11.20 - הודעה על הוספת קובץ txt המכיל את פונקציות העזר באתר הקורס