שפות תכנות, 234319

חורף 2018-2019

תרגיל בית 3

תאריך פרסום: 27/11/2018
מועד אחרון להגשה: 16/12/2018
מועד אחרון להגשה מאוחרת: 19/12/2018
מתרגל אחראי: טל שנקר
אי-מייל: tal.s@cs.technion.ac.il
בפניה בדוא"ל, נושא ההודעה (subject) יהיה "PL-EX3" (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

1. כתוב תכנית ב-C המדפיסה את הגודל בבתים של כל אחד מהטיפוסים היסודיים של שפת שיא המופיעים בשקפים. יתכן שיהיה עליך להעזר ב include

2. מצא את המפרט המאוחר ביותר של שפת ++C המופיע ברשת. צריף קישור. באילו כלים להגדרת השפה משתמש המפרט.

3. המפרט מספר שהטיפוס void הוא טיפוס לא מושלם שאי אפשר להשלים אותו. מצא את המקום שבו המפרט עושה זאת, תרגם לעברית את כל המונחים המופיעים במקום. (יהיה עליך אולי לעיין במקומות אחרים במפרט)

4. לגבי כל אחד מבין הישויות הבאות, ערך, פונקציה, משתנה, טיפוס, פרוצדורה, כתובת, ולגבי כל אחת משלוש השפות C ו ML ו פסקל (עליך ללמוד את השפה בהגדרה המופיעה כאן - http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/PascalRevisedReport(Wirth).pdf)

   a. זהה בשפת X מנגנון המאפשר יצירת ישות מסוג זה, ומתן שם עבורה. אם אין מנגנון מסוג זה, הסבר מדוע,

   b. זהה בשפת X מנגנון המאפשר יצירת ישות אנונימית מסוג זה, כלומר יצירת יישום מבלי שיש לישות הזו שם. אם אין מנגנון מסוג זה, מצא שפה אחרת מבין השפות שהוזכרו בקורס, והסבר את המנגנון הזה שם).

   c. זהה בשפת X מנגנון המאפשר מתן שם נוסף לישות קיימת מסוג זה. אם אין מנגנון מסוג זה, מצא שפה אחרת מבין השפות שהוזכרו בקורס, והסבר את המנגנון הזה שם).

חלק רטוב

תרגיל ראשון - One Liners.

בשאלה זו עליכם לממש את הפונקציות הבאות כ one liners. כלומר עליכם לממש את הפונקציות ללא שימוש בביטויי תנאי (if), ב pattern matching (כולל case of), קריאות רקורסיביות, ופונקציות עזר **(אין להשתמש גם ב let ו local)**. את הפתרון לתרגיל זה הגישו בקובץ lists.sml.

1. ממשו את האופרטור at המקבל רשימה ואינדקס ומחזירה את האיבר באינדקס ה i ברשימה. (הבהרה: דוגמאות ההרצה מטה צריכות לעבוד עם הפתרון שלכם, מבלי להניח שנריץ בשבילכם שורות קוד מסוימות כמו למשל infix at)

```
val at = fn : 'a list * int -> 'a
- [1, 2, 3, 4] at 6;
uncaught exception Subscript [subscript out of bounds]
- [1, 2, 3, 4] at 3;
val it = 4 : int
- [1, 2, 3, 4] at ~1;
uncaught exception Subscript [subscript out of bounds]
```

2. ממשו את הפונקציה enumerate המקבלת רשימה ומחזירה רשימה של tuples מהצורה (i, x_i) כאשר x_i הוא האיבר במקום ה i ברשימת הקלט.

```
val enumerate = fn : 'a list -> (int * 'a) list
- enumerate ["A", "B", "C", "D"];
val it = [(0,"A"),(1,"B"),(2,"C"),(3,"D")] : (int * string) list
- enumerate [];
(* It's okay to ignore Warning: type vars not generalized … *)
val it = [] : (int * ?.X1) list
```

3. ממשו את הפונקציה reverse המקבלת רשימה ומחזירה רשימה חדשה בה האיברים נמצאים בסדר הפוך.

```
val it = fn : 'a list -> 'a list
- reverse [false, true, true, false ,false];
val it = [false, false, true, true, false] : bool list
- reverse [];
(* It's okay to ignore Warning: type vars not generalized … *)
val it = [] : ?.X1 list
```

4. ממשו את הפונקציה flatten המקבלת רשימה של רשימות ומחזירה רשימה חדשה המכילה את האיברים של כל הרשימות הפנימיות על פי סדר הופעתם.

```
val flatten = fn : 'a list list -> 'a list
- flatten [[1, 2, 3], [4, 5, 6], [7, 8]];
val it = [1,2,3,4,5,6,7,8] : int list
- flatten [];
(* It's okay to ignore Warning: type vars not generalized … *)
val it = [] : ?.X1 list
- flatten [[], [1]];
val it = [1] : int list
```

5. ממשו את הפונקציה applyif המקבלת פונקציה, פרדיקט, ורשימה. הפונקציה מחזירה רשימה חדשה באותו הגודל בה מופיעים האיברים כפי שהם אם הם לא מקיימים את הפרדיקט ואחרת ההפעלה של פונקציית הקלט עליהם. **יש לשמוך על סדר האיברים המקורי.**

```
val applyif = fn : ('a -> 'a) -> ('a -> bool) -> 'a list -> 'a list
- applyif (fn x => x * x) (fn x => x > 5) [1, 3, 5, 7, 9, 11];
val it = [1,3,5,49,81,121] : int list
```

6. ממשו את הפונקציה slice המקבלת רשימה ו tuple של שני אינדקסים (s,e) ומחזירה רשימה חדשה המכילה את איברי הרשימה המקורית במקומות s עד e - 1 כולל.

```
val slice = fn : 'a list -> int * int -> 'a list
- slice [14, 7, 3, 1, 6, 1] (2, 4);
val it = [3,1] : int list
- slice [1] (2, 6);
uncaught exception Subscript [subscript out of bounds]
- slice [] (0, 0);
(* It's okay to ignore Warning: type vars not generalized … *)
val it = [] : ?.X1 list
```

7. ממשו את הפונקציה allholds המקבלת רשימה של פרדיקטים, ורשימה נוספת ומחזירה רשימה חדשה המכילה רק את האיברים ברשימה המקורית המקיימים את כל הפרדיקטים.

```
val allholds = fn : ('a -> bool) list -> 'a list -> 'a list
- allholds [] [1, 2, 3];
val it = [1,2,3] : int list
- allholds [fn x => x mod 2 = 0, fn y => y mod 3 = 0]
= [1, 3, 6, 8, 12, 18, 9];
val it = [6,12,18] : int list
```

תרגיל שני - חידות איינשטיין.

בתרגיל זה נרצה לכתוב תכנית ב ML לפתירת חידות איינשטיין. את הפתרון לחלק זה יש להגיש בקובץ einstein.sml. לא מכירים? קראו בקישור https://en.wikipedia.org/wiki/Zebra_Puzzle
אנו נייצג חידה בעזרת הטיפוסים הבאים (הוסיפו את ההגדרות לפתרון):

```
datatype characteristic = Char of string list;
```

כאשר מאפיין הוא למעשה רשימה של אפשרויות אשר הוא יכול לקבל (לדוגמה בקישור: צבע הבית).

```
datatype puzzle = Puzzle of characteristic list * (db -> db)
     withtype db = string list list;
```

חידה מורכבת מרשימה של מאפיינים ופונקציית פתרון אשר בהינתן רשימה של פתרונות אפשריים מחזירה רק את אלו המקיימים את כל העובדות הקיימות במערכת עד כה.

```
datatype connection = Connection of string list;
```

קישור הוא נתון לגבי הפתרון הנכון. מכיל רשימה שארכה כמספר המאפיינים. הרשימה מייצגת קשר של ערכים של מאפיינים שונים זה לזה (למשל בקישור: צבע הבית ומוצא הדייר).

1. ממשו את הפונקציה riddle המקבלת מספר n ומחזירה puzzle ריק המיוצג באמצעות רשימה של n תכונות ריקות, ופונקציית פתרון שלא מבצעת סינון.

```
val riddle = fn : int -> puzzle

- riddle 4;
val it = Puzzle ([Char [],Char [],Char [],Char []],fn) : puzzle
- riddle 0;
val it = Puzzle ([],fn) : puzzle
```

2. ממשו את הפונקציה המקבלת רשימה של מאפיינים ומחשבת את המכפלה הקרטזית שלהם, כלומר את כל הצירופים האפשריים לפתרון החידה.

```
val product = fn : characteristic list -> string list list

- product [Char ["A", "B", "C"], Char ["D", "E"]];
val it =
      [["A","D"],["A","E"],["B","D"],
      ["B","E"],["C","D"],["C","E"]] : string list list
- product [Char ["A", "B"], Char ["C", "D"], Char ["E"]];
val it =
      [["A","C","E"],["A","D","E"],
      ["B","C","E"],["B","D","E"]] : string list list
```

```
- product [Char ["A", "B"], Char nil];
val it = [] : string list list
```

3. כתבו את הפונקציה update המקבלת רשימה תכונות וקישור ומחזירה רשימת תכונות מעודכנת, ללא כפילויות. סדר התכונות בשני הארגומנטים זהה. במידה ותכונה מסוימת אינה חלק מהקישור יופיע במקום המתאים ברשימה מחרוזת ריקה.

```
val update = fn : characteristic list -> connection
                                        -> characteristic list

- [Char nil, Char nil, Char nil];
val it = [Char [],Char [],Char []] : characteristic list
- update it (Connection ["green", "", "milk"]);
val it = [Char ["green"],Char [],Char ["milk"]]
     : characteristic list
- update it (Connection ["", "dog", "milk"]);
val it = [Char ["green"],Char ["dog"],Char ["milk"]]
     : characteristic list
- update it (Connection ["red", "cat", ""]);
val it = [Char ["red","green"],Char ["cat","dog"],Char ["milk"]]
     : characteristic list
- update it (Connection ["blue", "", ""]);
val it = [Char ["blue","red","green"],
     Char ["cat","dog"],Char ["milk"]]
     : characteristic list
```

4. ממשו את הפונקציה trueConnection המקבלת חידה וקישור שידוע להיות נכון (כלומר צריך להתקיים בפתרון) ומחזירה חידה מעודכנת (גם פונקציית הפתרון).

```
val trueConnection = fn : puzzle -> connection -> puzzle
```

5. ממשו את הפונקציה falseConnection המקבלת חידה וקישור שידוע להיות לא נכון (כלומר לא יתקיים בפתרון החידה) ומחזירה חידה מעודכנת (גם פונקציית הפתרון).

```
val falseConnection = fn : puzzle -> connection -> puzzle
```

6. ממשו את הפונקציה solve המקבלת חידה ומחזירה רשימה של כל הצירופים האפשריים של מאפיינים בהינתן האילוצים של החידה, אם קיימים מספיק אילוצים, נקבל את פתרון החידה.

```
val solve = fn : puzzle -> string list list
```

לדוגמה חידה המבוססת על זו שכאן:

```
Control.Print.printDepth := 100;
infix ++;
fun solution ++ connection = trueConnection solution connection;
infix +!;
fun solution +! connection = falseConnection solution connection;

- riddle 5;
- it ++ (Connection ["Brit", "Red", "", "", ""]);
- it ++ (Connection ["Swede", "", "Dogs", "", ""]);
- it ++ (Connection ["Dane", "", "Horses", "Tea", ""]);
- it ++ (Connection ["", "Green", "", "Coffee", ""]);
- it ++ (Connection ["", "White", "", "Beer", "BlueMaster"]);
- it ++ (Connection ["Norwegian", "", "", "", "Dunhill"]);
- it ++ (Connection ["", "", "Birds", "Milk" ,"Pall Mall"]);
- it ++ (Connection ["German", "", "", "", "Prince"]);
- it ++ (Connection ["", "White", "", "", ""]);
- it ++ (Connection ["", "", "Fish", "", ""]);
- it +! (Connection ["", "", "Cats", "", "Blend"]);
- it +! (Connection ["Norwegian", "Blue", "", "", ""]);
- it +! (Connection ["", "Yellow", "Horses", "", ""]);
- it +! (Connection ["German", "", "Cats", "", ""]);
- it +! (Connection ["", "", "", "Water", "Prince"]);
- solve it;

val it =
  [["German","Green","Fish","Coffee","Prince"],
   ["Norwegian","Yellow","Cats","Water","Dunhill"],
   ["Dane","Blue","Horses","Tea","Blend"],
   ["Swede","White","Dogs","Beer","BlueMaster"],
   ["Brit","Red","Birds","Milk","Pall Mall"]] : string list list
```

תרגיל שלישי - בניית השפה Lisp

תרגיל זה הועבר לתרגיל בית אחר, ואין להגיש אותו (יכולים להיות שינויים בנוסח ובפונקציות המבוקשות).

בשאלה זו נכתוב בשפת ML פרשן של שפת MLISP. שפת MLISP היא גירסה פשטנית של שפת LISP. את הפתרון לחלק זה יש להגיש בקובץ mlisp.sml.

ראשית, נגדיר את טיפוס הערכים של MLISP. הטיפוס יכיל שלושה ערכים אטומיים:

- מחרוזת (שתוגדר באמצעות הבנאי STR)
- מספר שלם (שתוגדר באמצעות הבנאי INT)
- הקבוע הסימבולי NIL (באותיות גדולות, כאן ולהלן, אלא אם נאמר אחרת).

ערכים מורכבים יהיו CONS (שם הבנאי) של זוג סדור שבו הראשון הוא ה CAR והשני הוא ה CDR.

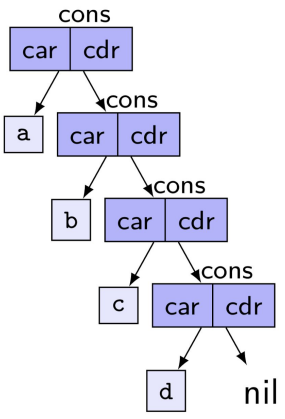
1. הגדירו טיפוס בשפת ML המתאר את טיפוס הערכים הללו. שם הטיפוס הזה יהיה S (קיצור ל SEXPRESSION או "ביטוי סימבולי")

2. הגדירו שם T לקבוע מהטיפוס שהוגדר בסעיף 1, שמכיל את הסדרית "T" .

3. הגדירו את הפונקציות האונריות הבאות ב MLISP שחתימת כולן תהיה: `fn: S -> S`

- CAR - הפונקציה המקבלת CONS ומחזירה את ה CAR אחרת זורקת חריגה.
- CDR - הפונקציה המקבלת CONS ומחזירה את ה CDR אחרת זורקת חריגה.
- NULL - הפונקציה מקבלת ביטוי סימבולי ומחזירה T אם הוא NIL ו NIL אחרת.
- INTEGER - הפונקציה מקבלת ביטוי סימבולי ומחזירה T אם הוא INT ו NIL אחרת.
- QUOTE - הפונקציה מחזירה את הארגומנט שקיבלה מבלי לשערך אותו.
- LST - הפונקציה מקבלת ביטוי סימבולי ומחזירה T אם הביטוי מהווה רשימה חוקית ו NIL אחרת. רשימה חוקית היא בעלת המבנה הבא: ה-CAR ברשימה צריך להיות איבר אטומי (כולל NIL) או רשימה בפני עצמו (אך לא איבר מורכב שאינו רשימה) וה-CDR צריך להיות רשימה גם באופן רקורסיבי, שימו לב שגם NIL נחשב רשימה (רשימה ריקה)
שתי הפונקציות הראשונות צריכות לזרוק exception אם יש שגיאה (השם לבחירתכם)

In binary tree representation:



4. הגדירו את הפונקציות הבינאריות הבאות עבור MLISP שחתימת כולן תהיה: `fn: S*S -> S`

- EQ - הפונקציה מבצעת השוואה רקורסיבית של שני ביטויים סימבוליים. מחזירה T במידה והם שווים ו NIL אחרת.
- PLUS - הפונקציה מחזירה ביטוי סימבולי עבור הסכום של השניים שקיבלה. הפונקציה זורקת חריגה במידה ואחד הארומגנטים שקיבלה אינו INT.
- TIMES - הפונקציה מחזירה ביטוי סימבולי עבור המכפלה של השניים שקיבלה. הפונקציה זורקת חריגה במידה ואחד הארומגנטים שקיבלה אינו INT.
- MEANING - הפונקציה מקבלת שני ביטויים סימבוליים כאשר הראשון שייקרא IDENTIFIER הוא STR והשני הוא רשימה (בייצוג LISP) של CONS (זוג סדור) המהווים BINDING. בכל אחד מה CONS האיבר הראשון הוא STR שמהווה IDENTIFIER והאיבר השני ביטוי S המהווה את המשמעות שלו. הפונקציה מחפשת את ה IDENTIFIER המתאים ל STR שהפונקציה קיבלה ומחזירה את ה BINDING שלו. אם לא נמצא ברשימה איבר מתאים הפונקציה מחזירה NIL. שימו לב שהרשימה אינה חוקית על פי ההגדרה של LST. מספר דוגמאות בכתיב LISP:

```
(search ‘foo ‘( a.12 a.f foo.(x y) foo.(a b) b.a)): returns (x y)
(search ‘b  ‘( a.12 a.f foo.(x) foo.(a b) b.a)): returns a
(search ‘ab  ‘( a.12 a.f foo.(x) foo.(a b) b.a)): returns nil
```

5. ממשו את הפונקציה הטרנרית COND עבור MLISP שחתימתה תהיה: `fn: S*S*S -> S` הפונקציה מחזירה את האגורמנט השני אם הראשון אינו NIL, אחרת מחזירה את השלישי.

6. הגדירו טיפוס חדש שהוא האיחוד של טיפוסי הפונקציות מסעיפים 3-5, שמות הבנאים יהיו UNARY, BINARY, TRINARY.

7. כעת, הרחיבו את ההגדרה של סעיף 1, כך שערכים אטומיים של הטיפוס S יוכלו להיות גם אחת הפונקציות שהוגדו בסעיפים 3-5. (עשו זאת ע״י הוספת בנאי בודד שייקרא SF להגדרה מסעיף 1)

8. הגדר מהדורה ראשונה של הפונקציה EVAL (את המהדורה השנייה נראה בתרגיל הבא). פונקציה זו תקבל שני ארגומנטים ביטוי לשיערוך, וסביבה, שהיא רשימה שמוגדרת כמתואר בעבור MEANING, ותפעל באופן רקורסיבי, כדי לשערך את הביטוי.

- ערכו של NIL הוא NIL.
- ערכו של INTEGER הוא הערך שלו עצמו.
- ערכו של STRING הוא תוצאת החיפוש של ה STRING בסביבה.
- ערכה של רשימה יחושב כך:

i. ראשית נשערך את האיבר הראשון. אם האיבר הראשון אינו פונקציה כפי שהוגדרה בסעיף 7, זוהי שגיאה ונזרוק חריגה.
ii. שאר האיברים ברשימה יהיו הארגומנטים לפונקציה, אם מספר הארגומנטים אינו תואם בדיוק לזה שהפונקציה מצפה, נזרוק חריגה.
iii. נשערך את שאר האיברים ברשימה באופן רקורסיבי.
iv. נפעיל את הפונקציה עליהם - תוצאת ההפעלה היא התוצאה של הפונקציה.

9. תקנו את המימוש של הקוד כך שאם EVAL נתקלה בפונקציה QUOTE לא יתבצע שערוך של ה הערך המתאים לו. רמז: אפשר לשנות את מבנה הנתונים.

הנחיות

- בתרגיל זה ניתן להשתמש רק בחומר שנלמד בשפת ML עד תרגול 6. אין להשתמש באף פונקציה או תכונה של השפה שלא נלמדה בתרגולים.
- מותר לכם להגדיר פונקציות נוספות כרצונכם אך עליכם **להסתיר אותן** באמצעות let/local.
- רשימת הקבצים שצריכים להופיע בתוך קובץ ה-zip היא:

```
dry.pdf, einstein.sml, lists.sml
```

- יש להגיש את הקבצים דחוסים בתוך קובץ zip. הקבצים יהיו בשורש קובץ ה-zip ולא בתוך ספרייה. שם הקובץ יהיה **EX3_ID1_ID2.zip** כאשר ID1,ID2 הם מספרי ת.ז. של המגישים.
- שימו לב שהבדיקה של החלק הרטוב היא אוטומטית, ולכן הקפידו על מילוי כל ההוראות בשביל למנוע בעיות מיותרות.
- בודקי התרגילים אוהבים Memes. לאור ההצלחה בסמסטר הקודם, גם הפעם, שתפו את תחושותיכם במהלך פתירת התרגיל באמצעות Meme מתאים על דף השער בהגשה.

בהצלחה!

תיקונים והבהרות

1. בתרגיל אחד בחלק הרטוב נוספה הבהרה כי אסור להשתמש ב let ו local.
2. בתרגיל 3 בחלק הרטוב - הוחלף SEARCH ב MEANING.
3. בתרגיל 1 בחלק הרטוב - סעיף 5. נוספה הערה.
4. תרגיל 3 בחלק הרטוב הועבר לתרגיל בית אחר ואין להגישו.
5. תרגיל 2 בחלק הרטוב - נוספו דוגמאות הרצה עבור riddle, product, update.
6. בהוראות ההגשה תוקן שם ה zip מ EX2 ל EX3.