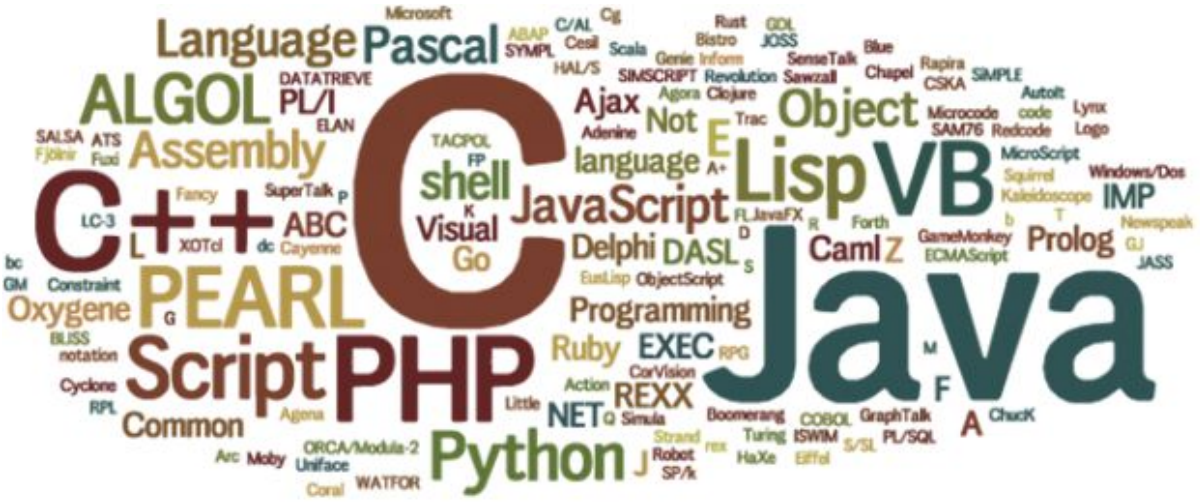
שפות תכנות, 236319

חורף 2018-2019



תרגיל בית 5

תאריך פרסום: 30/12/2018
מועד אחרון להגשה: 10/01/2019
מועד אחרון להגשה מאוחרת: 13/01/2019
מתרגל אחראי: יוסי גורשומוב
אי-מייל: yossigor@campus.technion.ac.il
בפניה בדוא"ל, נושא ההודעה (subject) יהיה "PL-EX5" (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

שאלה 1 RUST:

א)

בשפת Rust קיים מנגנון ownership המסייע בניהול הזיכרון. מתי משתחרר זיכרון שהוקצה בשפת Rust, ומה בקוד יכול לשנות את זמן השחרור של הזיכרון בזמן ריצת התוכנית? הניחו שבקוד לא נעשה שימוש במילת המפתח unsafe (הערה: על מילת המפתח unsafe לא למדנו ולכן יש להתעלם מאפשרות זו).

ב)

בשפת Rust יש גם מנגנון borrow checker. מתי חוקי בשפת Rust ליצור reference למשתנה, כלומר לעשות borrow?

בשני הסעיפים הבאים נתון קוד RUST שאינו מתקמפל. בצעו מספר של תיקונים מינימלי על מנת שהוא יבצע את המטרה שלו. לכל תיקון שהוספתם ציינו:

- מה גרם לבעיה
- כיצד התיקון שלכם פתר אותה.

התייחסו בתשובתכם למושגים שנלמדו בתרגול, תשובה ללא הסבר לא תתקבל.

ג)

על הקוד להדפיס את המספרים 1-9 פעמיים.

```rust
fn main(){
   let vec = Vec::new();
   for i in 1..10 {
       vec.push(i);
   }
   for i in vec{
       println!("{}",i);
   }
   for i in vec{
       println!("{}",i);
   }
}
```

ד)

על הקוד להדפיס `"The first element is:1"`

```rust
fn main(){
   let mut v = vec![1, 2, 3, 4, 5];
   let first = &v[0];
```

```
    v.push(6);
    println!("The first element is: {}", first);
}
```

ה)

נתון קטע הקוד הבא בשפת Rust. הפונקציה fold פועלת באופן דומה ל foldl.List שלמדנו בשפת ML.
הפרמטר השני לפונקציה fold הוא פונקציה אנונימית. החוקים של הפונקציות האנונימיות ב-Rust דומים לחוקים של הפונקציות האנונימיות ב-C++11 . הסבירו מדוע בהגדרת הפונקציה האנונימית מועבר b כרפרנס (מושאל) ו-a לא?

```
fn sum_vec(v: &Vec<i32>) -> i32 {
    v.iter().fold(0, |a, &b| a + b)
}
```

הבהרה: בהגדרת הפונקציה fold הפרמטר השני לפונקציה האנונימית מוגדר להיות reference. לכן גם אם תסירו את סימן הreference מb. עדיין תתקיים השאלה של הערך, כלומר b כעת יהיה מטיפוס i32&. האופרטור + אמנם הוא סלחן ומסוגל לקבל גם פרמטרים מטיפוס i32& וגם מטיפוס i32 ולהחזיר ערך מטיפוס i32. **לכן אין צורך שתחפשו כיצד לשבור את הקוד ולגרום לכך שלא יהיה ניתן להשתמש בvector לאחר השימוש בפונקציה אלא יש לענות על השאלה למה בהגדרת הפונקציה fold הפרמטר השני לפונקציה האנונימית מועבר כרפרנס.**

דוגמאת קוד שאולי תבהיר את הסיפור:

```
fn add (a:i32, b:i32) -> i32{
    return a+b;
}
fn sum_vec(v: &Vec<i32>) -> i32 {
   v.iter().fold(0, |a, b| add(a,b))
}
fn main(){
    let v = Vec::new();
    v.push(1);
    v.push(2);
    let x = sum_vec(&v);
    for i in &v{
        println!("{}",i);
    }
}
```

כאן החלפנו את השימוש באופרטור + ואם נריץ את הקוד נקבל שגיאה שb מטיפוס i32& ולכן לא נוכל להפעיל את הפונקציה add. (אם נוסיף את הסימן & לb אז b יהיה מטיפוס i32 והקוד יפעל כרגיל).

שאלה 2 חזרה על איחסון:

מהי שיטת ניהול הזכרון בשפת נים? האם יש איסוף אשפה? אריתמטיקה של מצביעים? משתני מחסנית הגוועים מעצמם? כיצד מיוצגים מערכים? מה קורה בחריגה מגבולות מערך? **קצת חזרה על החומר שנלמד בשיעור, וקצת חיפוש בויקיפדיה ובגוגל. עליכם ל"זהות" את המונחים "אריתמטיקה של מצביעים", ו"משתני מחסנית 'גוועים' מעצמם". לא סיפור גדול.**

שאלה 3 - RTTI

1. קראו את פרק 5.6 בחוברת השקפים בנושא RTTI.
   מהו מנגנון RTTI? סכמו בשני משפטים.
2. האם בשפת C יש מנגנון RTTI?
   אם כן - תארו אותו.
   אם לא - הסבירו למה.
3. הסבירו: כיצד מנגנון איסוף אשפה משתמש במנגנון RTTI?
4. תארו שימוש אחר (שאינו איסוף אשפה) למנגנון RTTI.

שאלה 4 - פקודות

קראו את השקפים של פרק 6 וענו על השאלות הבאות בנוגע לשפה Rust:

1. מהם הפקודות האטומיות בשפה?
2. מהם בנאי הפקודות בשפה?
3. מהם ה Sequencers של השפה?

חלק רטוב

שאלה 1: רשימות מתחלפות

רקע: הטיפוס 'a list בשפת ML מוגדר כך:

```
datatype 'a list = nil | :: of 'a * 'a list
infixr 5 ::
```

(infixr הוא infix עם אסוציאטיביות ימנית. 5 הוא ספרה שמגדירה את העדיפות בין אופרטורים.)

פרט לכך, ML מגדירה תחביר מיוחד עבור רשימות, בעזרת סוגריים מרובעים:

```
[1,2,3] = 1::2::3::nil
```

החיסרון של רשימות ביחס ל tuple, למשל, היא שכל האיברים של רשימה נתונה הם מאותו טיפוס.

בתרגיל זה ננסה לתקן את המצב.

1. הגדירו טיפוס גנרי "רשימה מתחלפת". ערכים מטיפוס זה מייצגים רשימות המחזיקות איברים מטיפוס 'a, אחריו איבר מטיפוס 'b, אחריו איבר מטיפוס 'a וכן הלאה. רשימות עשויות להיות בכל אורך שהוא, כולל 0.

   טיפוס שיאפשר לבצע את המשימות הנדרשות בהמשך השאלה יקיים את הדרישה.

   השלימו את התבנית עבור הטיפוס והוסיפו את ההגדרה לקובץ הפתרון:

```
datatype ('a, 'b) heterolist = ____________________
                             | ____________________
```

   אין לחרוג מצורה זאת - על ההגדרה להכיל רק תו | בודד. אין להשתמש בטיפוס list.

2. בשפת ML לא ניתן להגדיר תחביר מקביל העושה שימוש בסוגריים מרובעים, אך אם היה ניתן להגדיר תחביר כזה היה אפשר ליצור רשימות כגון:

```
val x1y2 : (string, int) heterolist  = ["x",1,"y",2]
```

   הגדירו את הפונקציה:

```
build4 : 'a*'b*'a*'b -> ('a, 'b) heterolist
```

   שמחזירה רשימה מתחלפת, המחזיקה בדיוק את ארבעת הערכים שהועברו בפרמטר.

   הפתרון צריך להינתן בביטוי בודד (אין להגדיר פונקציות עזר פנימיות וכו'):

```
fun build4 (x,one,y,two) = _____________
```

3. **(לא להגשה)**: השלימו את התבנית בהצהרה להלן כך שכל אחד מהמשתנים x, one, y, two יחזיק את הערך המתאים לשמו, ושתי השורות יתקמפלו וירוצו ללא חריגה (כאשר הן מבוצעות ברצף):

```
val _______________ = build4 ("x",1,"y",2)
val ("x",1,"y",2) = (x,one,y,two)
```

מומלץ מאוד לא להמשיך לסעיפים הבאים לפני שהצלחתם את החלקים הקודמים.

4. הגדירו פונקציה בחתימה להלן (במדויק):

```
unzip : ('a, 'b) heterolist  -> 'a list * 'b list
```

על כל רשימה בזוג המוחזר להחזיק בדיוק את כל האיברים מהטיפוס המתאים ברשימה שהתקבלה כפרמטר, ולפי אותו סדר. פונקציה זאת תמיד תחזיר ערך (לא ייזרקו חריגות) בפרט, על השורה הבאה להתקמפל ולרוץ ללא שגיאות וללא הודעות מיוחדות:

```
val (["x","y"], [1,2]) = unzip (build4 ("x",1,"y",2))
```

5. הגדירו פונקציה `zip : 'a list * 'b list -> ('a, 'b) heterolist` המבצעת את הפעולה ההפוכה לפונקציה בסעיף הקודם. במקרה שמספר האיברים ברשימות איננו מתאים (אם לא מתקיים ש`len(a) - 1 ≤ len(b) ≤`□□□ `len(a)`) להלן (שאיננה `id_heterolist` הפונקציה .Empty יש לזרוק את החריגה `a, 'b) heterolist'):` להגשה) שקולה לפונקציית הזהות על

```
val id_heterolist  = zip o unzip (* ignore the warning *)
```

וכן הקוד הבא צריך להתקמפל ולרוץ ללא שגיאות וללא הודעות מיוחדות:

```
val (["x","y"], [1,2]) = unzip (zip (["x", "y"], [1,2]))
```

שאלה 2: רצפים

לכל אורך השאלה, נשתמש בהגדרה הבאה של רצף (כפי שנלמד בתרגולים):

```
datatype 'a seq = Nil | Cons of 'a * (unit-> 'a seq);
```

ובהגדרות הבאות של פונקציות head/tail (שהן שונות קצת מאלו שהוצגו בתרגול):

```
exception EmptySeq;
fun head(Cons(x,_)) = x | head Nil = raise EmptySeq;
fun tail(Cons(_,xf)) = xf() | tail Nil = raise EmptySeq;
```

עליכם להוסיף את ההגדרות הללו בתחילת קובץ הפתרון.

בשאלה זו נעסוק בהרחבה של טיפוס הרצף שנלמד בכיתה, הנקראת "רצף דו-כיווני".
רצף כזה הוא רצף שניתן להתקדם בו קדימה ואחורה. לשם כך, נגדיר את שני ה- datatypes הבאים:

```
datatype direction = Back | Forward;
datatype 'a bseq =   bNil
                   | bCons of 'a * (direction -> 'a bseq);
```

כאשר bNil מייצג את הרצף הריק ו- bCons מייצג איבר ברצף, המכיל ערך ופונקציית התקדמות.
פונקציה זו מקבלת ערך מטיפוס direction (Back או Forward) ומחזירה את האיבר הקודם או הבא ברצף, בהתאמה.
נגדיר בנוסף את שלושת הפונקציות הבאות:

```
fun bHead(bCons(x,_)) = x | bHead bNil = raise EmptySeq;
fun bForward(bCons(_,xf)) = xf(Forward) | bForward bNil =
raise EmptySeq;
fun bBack(bCons(_,xf)) = xf(Back) | bBack bNil = raise
EmptySeq;
```

יש להוסיף את ה datatypes והפונקציות שהוגדרו לעיל לקובץ הפתרון שלכם.

1. ממשו את הפונקציה `intbseq : int -> int bseq`, המקבלת מספר שלם x ומחזירה "רצף דו-כיווני" אינסופי המייצג את כל המספרים השלמים, כאשר האיבר הנוכחי ברצף המוחזר מכיל את הערך x. לכל איבר y ברצף, האיבר העוקב לו הוא y+1 והאיבר הקודם לו הוא y-1 .

דוגמת הרצה:

```
- intbseq 2;
val it = bCons (2,fn) : int bseq
- bForward(it);
val it = bCons (3,fn) : int bseq
- bForward(it);
val it = bCons (4,fn) : int bseq
- bBack(it);
val it = bCons (3,fn) : int bseq
- bBack(it);
val it = bCons (2,fn) : int bseq
- bBack(it);
val it = bCons (1,fn) : int bseq
- bBack(it);
val it = bCons (0,fn) : int bseq
- bBack(it);
val it = bCons (~1,fn) : int bseq
```

2. ממשו את הפונקציה `bmap : ('a -> 'b) -> 'a bseq -> 'b bseq`, המקבלת פונקציה f ו"רצף דו-כיווני", ומחזירה "רצף דו-כיווני" שבו כל איבר עם ערך x הופך לאיבר עם ערך f(x).

דוגמת הרצה:

```
- bmap (fn x =>x*x) (intbseq 2);
val it = bCons (4,fn) : int bseq
- bForward(it);
val it = bCons (9,fn) : int bseq
- bForward(it);
val it = bCons (16,fn) : int bseq
```

```
- bBack(it);
val it = bCons (9,fn) : int bseq
- bBack(it);
val it = bCons (4,fn) : int bseq
- bBack(it);
val it = bCons (1,fn) : int bseq
- bBack(it);
val it = bCons (0,fn) : int bseq
- bBack(it);
val it = bCons (1,fn) : int bseq
- bBack(it);
val it = bCons (4,fn) : int bseq
```

3. ממשו את הפונקציה:

```
bfilter : ('a -> bool) -> direction -> 'a bseq -> 'a bseq
```

המקבלת פונקציית פרדיקט ,איבר d מטיפוס direction ו"רצף דו-כיווני". הפונקציה מחזירה "רצף דו-כיווני" המכיל רק איברים עם ערכים שהפרדיקט מחזיר עבורם true. אם האיבר הנוכחי ברצף שהתקבל אינו ברצף המוחזר, אזי האיבר הנוכחי ברצף המוחזר יהיה האיבר הקרוב ביותר מכיוון d שהפרדיקט מחזיר עבורו true (כלומר, אם d=Forward, יוחזר "האיבר הבא" הקרוב אליו ביותר שהפרדיקט מחזיר עבורו true, אחרת יוחזר "האיבר הקודם" הקרוב אליו ביותר שהפרדיקט מחזיר עבורו true). אם לא נמצא איבר בכיוון d שהפרדיקט מחזיר עבורו true והרצף הוא אינסופי תכנס הפונקציה ללולאה אינסופית.

דוגמת הרצה:

```
- bfilter (fn x => x mod 2 = 0) Back (intbseq 2);
val it = bCons (2,fn) : int bseq
- bForward(it);
val it = bCons (4,fn) : int bseq
- bForward(it);
val it = bCons (6,fn) : int bseq
- bBack(it);
val it = bCons (4,fn) : int bseq
- bBack(it);
val it = bCons (2,fn) : int bseq
- bBack(it);
val it = bCons (0,fn) : int bseq
```

```
- bBack(it);
val it = bCons (~2,fn) : int bseq
- bfilter (fn x => x mod 2 = 0) Back (intbseq 1);
val it = bCons (0,fn) : int bseq
- bfilter (fn x => x mod 2 = 0) Forward (intbseq 1);
val it = bCons (2,fn) : int bseq
```

4. ממשו את הפונקציה `seq2bseq : 'a seq -> 'a seq -> 'a bseq`, המקבלת שני רצפים חד-כיווניים, כאשר לרצף הראשון נקרא הרצף ההפוך ולשני נקרא הרצף הרגיל. הפונקציה תחזיר "רצף דו-כיווני", כאשר הערך של האיבר הנוכחי ברצף המוחזר הוא הערך של האיבר הראשון ברצף הרגיל (הפרמטר השני שהתקבל) כל האיברים הבאים אחרי אותו איבר הם האיברים הבאים אחריו ברצף הרגיל. כל האיברים הקודמים לו הם האיברים המופיעים ברצף ההפוך, לפי סדר הופעתם (כלומר, האיבר שקודם לאיבר הנוכחי הוא האיבר הראשון ברצף ההפוך, האיבר שלפניו הוא האיבר השני ברצף ההפוך וכן הלאה). ניתן להניח ששני הרצפים המתקבלים הם אינסופיים, ולכן אין צורך לטפל במקרה שמתקבל Nil.

דוגמת הרצה:

נגדיר את שתי הפונקציות הבאות, המחזירות רצפים חד-כיווניים המכילים את כל המספרים השלמים בסדר עולה ויורד, בהתאמה (אין צורך להוסיף את הגדרות הפונקציות הללו לקובץ ההגשה) :

```
fun from(x) = Cons(x, fn()=>from(x+1));
fun downfrom(x) = Cons(x, fn()=>downfrom(x-1));
```

כעת נגדיר את הרצף הדו-כיווני המכיל את כל המספרים השלמים, השקול לרצף שהגדרנו בסעיף א' (כאשר האיבר ההתחלתי הוא 0) באופן הבא:

```
-seq2bseq (downfrom ~1) (from 0);
val it = bCons (0,fn) : int bseq
```

כעת ניתן להקיש את רצף הפקודות הבא, ולקבל את הפלטים הרשומים:

```
-bForward(it);
val it = bCons (1,fn) : int bseq
-bBack(it);
val it = bCons (0,fn) : int bseq
-bBack(it);
val it = bCons (~1,fn) : int bseq
```

5. ממשו את הפונקציה `bSeqJump : 'a bseq -> int -> 'a bseq` המקבלת "רצף דו-כיווני" אינסופי ומספר שלם וחיובי m, ומחזירה "רצף דו-כיווני" המכיל רק את האיברים מרצף הקלט

שהאינדקס של המיקום שלהם מתחלק ב-m ללא שארית. למשל, אם m=2 אז הרצף המוחזר יכיל רק את האיברים מהרצף המקומי הנמצאים במקומות זוגיים. ניתן להניח שהרצף המתקבל הוא אינסופי ושהמספר m הוא חיובי ממש (גדול מאפס) ואין צורך לבדוק זאת. בנוסף הניחו שהמיקום של האיבר ההתחלתי של רצף הקלט הוא 0.

```
- intbseq 0;
val it = bCons (0,fn) : int bseq
- bSeqJump it 3;
val it = bCons (0,fn) : int bseq
- bForward it;
val it = bCons (3,fn) : int bseq
- bForward it;
val it = bCons (6,fn) : int bseq
- bBack it;
val it = bCons (3,fn) : int bseq
- bBack it;
val it = bCons (0,fn) : int bseq
- bBack it;
val it = bCons (~3,fn) : int bseq
```

הנחיות הגשה

- בתרגיל זה ניתן להשתמש רק בחומר שנלמד בתרגולים. אין להשתמש באף פונקציה או תכונה של השפה שלא נלמדה בתרגולים.

- רשימת הקבצים שצריכים להופיע בתוך קובץ ה-zip היא:

```
dry.pdf, sol.sml
```

- לכל קובץ שמכיל קוד הוסיפו בשורה הראשונה הערה המכילה את השם, מספר ת.ז. וכתובת המייל של המגישים מופרדים באמצעות רווח. וודאו שהקבצים נטענים/מתקמפלים ללא שגיאות גם לאחר הוספת ההערה שלעיל (**<u>ובמיוחד לפני ההגשה!</u>**)

- יש להגיש את הקבצים דחוסים בתוך קובץ zip. הקבצים יהיו בשורש קובץ ה-zip ולא בתוך ספרייה. שם הקובץ יהיה EX5_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.

בהצלחה!