ה א ו נ י ב ר ס י ט ה ה פ ת ו ח ה

20905

שפות תכנות

חוברת הקורס- אביב 2021ב

כתב: דני כלפון

פברואר 2021 - סמסטר אביב – תשפ"א

פנימי – לא להפצה.

© כל הזכויות שמורות לאוניברסיטה הפתוחה.

תוכן העניינים

אל הסטודנט	א
1. לוח זמנים ופעילויות	ב
2. תיאור המטלות	ד
2.1 מידע כללי	ד
2.2 מבנה המטלות	ד
3. התנאים לקבלת נקודות זכות	ה
ממ"ן 11	1
ממ"ן 12	3
ממ"ן 13	7
ממ"ן 14	9
ממ"ן 15	11
ממ"ן 16	15

אל הסטודנט,

אנו מקדמים את פניך בברכה עם הצטרפותך אל הלומדים בקורס "שפות תכנות". בחוברת זו תמצא לוח הזמנים של הקורס, מטלות ותנאים לקבלת נקודות זכות בקורס.

לקורס קיים אתר באינטרנט בו תמצאו חומרי למידה נוספים, אותם מפרסם/מת מרכז/ת ההוראה. בנוסף, האתר מהווה עבורכם ערוץ תקשורת עם צוות ההוראה ועם סטודנטים אחרים בקורס. פרטים על למידה מתוקשבת ואתר הקורס, תמצאו באתר שה"ם בכתובת:

http://telem.openu.ac.il

מידע על שירותי ספרייה ומקורות מידע שהאוניברסיטה מעמידה לרשותכם, תמצאו באתר הספריה באינטרנט www.openu.ac.il/Library.

שעות הייעוץ הטלפוני שלי יפורסמו סמוך לפתיחת הסמסטר באתר הקורס. אפשר ורצוי לפנות אלי בדואר אלקטרוני: dannyca@openu.ac.il, תוך ציון שם מלא, מספר תעודת זהות ומספר טלפון. פגישות חובה לתאם מראש. לצורך בירורים בנושאים אדמיניסטרטיביים יש לפנות בכתב או טלפונית למחלקות האוניברסיטה הפתוחה.

הקורס הוא תכנותי באופיו, וכולל מטלות תכנותיות החשובות להבנת החומר ותרגולו. מומלץ שתקדישו זמן ראוי ללמידת חומר הקורס, שכן זה קורס השונה באופיו מקורסים תכנותיים אחרים המוכרים לכם באו"פ. הקורס כולל פיתוח מפרשים לשפות תכנות פשוטות המדגימות עקרונות הקיימים בשפות תכנות מודרניות. השתתפות במפגשים, הקדשת זמן ראוי ללמידת החומר והגשת המטלות הם הדרך הנכונה לסיום הקורס בהצלחה.

בברכת לימוד מהנה

כלפון דני

מרכז ההוראה בקורס

1. לוח זמנים ופעילויות (20905 / ב2021)

שבוע לימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח ממ"ן (למנחה)
1	05.03.2021-28.02.2021	1	מפגש ראשון	
2	12.03.2021-07.03.2021	2		
3	19.03.2021-14.03.2021	2	מפגש שני	הגשת ממן 11 17.03.2021
4	26.03.2021-21.03.2021	2		
5	02.04.2021-28.03.2021 (א-ו פסח)	3	מפגש שלישי	
6	09.04.2021-04.04.2021 (ה יום הזכרון לשואה)	3		הגשת ממן 12 07.04.2021
7	16.04.2021-11.04.2021 (ד יום הזיכרון, ה יום העצמאות)	3	מפגש רביעי	
8	23.04.2021-18.04.2021	3		
9	30.04.2021-25.04.2021 (ו ל"ג בעומר)	3	מפגש חמישי	הגשת ממן 13 28.04.2021

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

לוח זמנים ופעילויות - המשך

שבוע הלימוד	תאריכי שבוע הלימוד	יחידת הלימוד המומלצת	מפגשי ההנחיה*	תאריך אחרון למשלוח הממ"ן (למנחה)
10	07.05.2021-02.05.2021	4		
11	14.05.2021-09.05.2021	4	מפגש שישי	הגשת ממן 14 12.05.2021
12	21.05.2021-16.05.2021 (ב שבועות)	4		
13	28.05.2021-23.05.2021	7	מפגש שביעי	הגשת ממן 15 26.05.2021
14	04.06.2021-30.05.2021	7		
15	11.06.2021-06.06.2021	7	מפגש שמיני	הגשת ממן 16 09.06.2021

מועדי בחינות הגמר יפורסמו בנפרד

** התאריכים המדויקים של המפגשים הקבוצתיים מופיעים ב"לוח מפגשים ומנחים".*

.

2. תיאור המטלות

לתשומת לבכם!

כדי לעודדכם להגיש לבדיקה מספר רב של מטלות הנהגנו את ההקלה שלהלן:

אם הגשתם מטלות מעל למשקל המינימלי הנדרש בקורס, **המטלות** בציון הנמוך ביותר, שציוניהן נמוכים מציון הבחינה (**עד שתי מטלות**), לא יילקחו בחשבון בעת שקלול הציון הסופי.

זאת בתנאי שמטלות אלה **אינן חלק מדרישות החובה בקורס** ושהמשקל הצבור של המטלות האחרות שהוגשו, מגיע למינימום הנדרש.

זכרו! ציון סופי מחושב רק לסטודנטים שעברו את בחינת הגמר בציון 60 ומעלה והגישו מטלות כנדרש באותו קורס.

שימו לב!

בקורס זה **חובה** להגיש את המטלות בזמן, בהתאם לתאריך ההגשה המצוין עליהן. במקרים חריגים, כאשר יש סיבה מוצדקת להגשת המטלה באיחור, יש לפנות **בכתב** בדואר אלקטרוני אל מרכז ההוראה בקורס. **את הבקשה יש להגיש מראש!** יש לצרף לבקשה אישורים רשמיים, להצדקת סיבת הבקשה.

מטלות שיוגשו באיחור ללא אישור יבדקו והציון שיוזן עבורן יהיה 0, ללא תלות בציון של הבדיקה.

שימו לב, טיפול בבקשות שנשלחות לאחר מועד ב׳ של הסמסטר אינו בסמכות מרכז ההוראה, ויש להפנותן את האחראית על פניות סטודנטים של החטיבה למדעי המחשב.

להלן פירוט הניקוד לכל מטלה:

ממ״ן	ניקוד
11	5
12	5
13	5
14	5
15	5
16	5

2.2 מבנה המטלות

כל מטלה מורכבת מכמה שאלות. בראש כל שאלה מצוין משקלה היחסי בקביעת ציון המטלה. אם השאלה בממ״ן אינה ברורה לך, אל תהסס להתקשר אל אחד המנחים (בשעות הייעוץ הטלפוני שלו) לצורך קבלת הסבר, או הפנה את שאלתך אל פורום הדיונים של הקורס.

3. התנאים לקבלת נקודות זכות

כדי לקבל נקודות זכות בקורס זה עליך לעמוד בדרישות הבאות:

א) **צבירת משקל של לפחות 20 נקודות במטלות.**

ב) ציון של לפחות 60 נקודות בבחינת הגמר.

ג) ציון סופי בקורס של 60 נקודות לפחות.

לתשומת לבכם:

מדיניות קורס זה היא לאשר הזנת ציון אפס במטלות שלא הוגשו כנדרש בקורס.

סטודנטים אשר לא הגישו את מכסת המטלות המינימאלית לעמידה בדרישות הקורס ולקבלת זכאות להיבחן, ומבקשים שמטלות חסרות יוזנו בציון אפס, יפנו למוקד הפניות והמידע בטלפון **09-7782222** או **יעדכנו בעצמם** באתר שאילתא http://www.openu.ac.il/sheilta **קורסים ⇦ ציוני מטלות ובחינות ⇦ הזנת ציון 0 למטלות רשות שלא הוגשו.**

יש לקחת בחשבון כי מטלות אשר יוזן להן ציון אפס ישוקללו בחישוב הציון הסופי ובכך יורידו ציון זה ולא ניתן יהיה להמירן במטלות חלופיות במועד מאוחר יותר. על כן קיימת אפשרות שסטודנט אשר יעבור את הבחינה בהצלחה ייכשל בקורס (כשהממוצע המשוקלל של המטלות והבחינה יהיה נמוך מ- 60).

כלל זה איננו חל על מטלות חובה או על מטלות שנקבע עבורן ציון מינימום.

מטלת מנחה (ממ"ן) 11

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 1 בספר הלימוד, פרק 3 במדריך הלמידה, חלקו השני של מדריך הלמידה העוסק בשפת scheme ובסביבת העבודה racket וכן מדריכים באתר הקורס.

מספר השאלות: 5 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **17.3.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

שימו לב, בכל מקום במטלה בו נכתבה המילה סמל, הכוונה היא ל-scheme symbol.

שאלה 1 (15 נקודות)

כתבו פרוצדורה בשם my_flatten המקבלת רשימה כפרמטר. הפרוצדורה מחזירה את תוצאת שיטוח הרשימה.

להלן דוגמא:

```
>(my_flatten  '( (1 2)  ((3))  (4 5 6)))
'(1 2 3 4 5 6)
```

שאלה 2 (30 נקודות)

א) כתבו פרוצדורה בשם my_count המקבלת כפרמטרים פרדיקט ורשימה.
הפרוצדורה מחזירה כמה איברים ברשימה מקיימים את הפרדיקט.
להלן דוגמה:

```
>(my_count  positive?  '(1 3 -7 9 -2))
3
```

את הפרוצדורה ממשו באמצעות רקורסיה.

ב) כתבו פרוצדורה בשם my_count_foldr המחזירה תוצאה זהה לזו של my_count , הפעם ממשו אותה באמצעות שימוש ב- foldr וללא שימוש ברקורסיה (כלומר, במימוש לא תופיע קריאה ישירה לפרוצדורה אותה אתם נדרשים לממש).

שאלה 3 (15 נקודות)

כתבו פרוצדורה בשם my_partition המקבלת פרדיקט ורשימה כפרמטרים. הפרוצדורה מחזירה רשימה המורכבת מ-2 תתי רשימות, האחת תכיל איברים שקיימו את הפרדיקט, והשניה איברים שלא קיימו את הפרדיקט.
לדוגמה:

```
> (my_partition even? '(1 2 3 4 5 6))
'((2 4 6) (1 3 5))
```

שאלה 4 (15 נקודות)

כתבו פרוצדורה בשם my_append_map הפועלת בצורה דומה לפרוצדורה append_map.
להלן דוגמה להמחשת ההבדל בין map לבין append-map:

```
> (map cdr '((1 2 3) (4 5 6) (7 8 9)))
'((2 3) (5 6) (8 9))
> (append-map cdr '((1 2 3) (4 5 6) (7 8 9)))
> '(2 3 5 6 8 9)
```

שאלה 5 (25 נקודות)

כתבו פרוצדורה בשם my_permutations המקבלת כפרמטר רשימה. הפרוצדורה תחזיר את קבוצת כל התמורות (פרמוטציות) של איברי הרשימה. התשובה המוחזרת תהיה רשימה של תתי רשימות, כאשר כל תת-רשימה מייצגת את אחת התמורות.
לדוגמה:

```
> (my_permutations '(1 2 3))
'((1 2 3) (2 1 3) (1 3 2) (3 1 2) (2 3 1) (3 2 1))
```

מטלת מנחה (ממ"ן) 12

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 2 בספר הלימוד, פרק 4 במדריך הלמידה.

מספר השאלות: 3 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **7.4.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

לצורך פתרון מטלה זו עליכם ללמוד היטב את פרק 2 בספר הלימוד בליווי של פרק 4 במדריך הלמידה.

שאלה 1 (50 נקודות)

כאשר אנו רוצים לייצג סדרה כלשהי של ערכים, נוח לעשות זאת באמצעות רשימה. הרשימה (list של שפת Scheme) מאפשרת להגיע לאיבר הבא ברשימה בצורה נוחה (cdr...) אך אינה מאפשרת לעבור לאיבר הקודם בצורה נוחה ללא שימוש בארגומנטים נוספים.

בשאלה זו ניישם ייצוג לסדרת מספרים דו-כיוונית כפי שמוגדרת ע"י הדקדוק הבא:

NodeInSequence ::= (Int Listof(Int) Listof(int))

הרשימה הראשונה של מספרים כוללת את האיברים בסדרת המספרים שנמצאים לפני האיבר הנוכחי (זה שמיוצג ע"י ה- Int הראשון בדקדוק). ואילו הרשימה השניה של מספרים כוללת את המספרים בסדרה שנמצאים אחרי האיבר הנוכחי. לדוגמא:

(6 (5 4 3 2 1) (7 8 9))

מייצג את הרשימה (1 2 3 4 5 6 7 8 9) כאשר האיבר הנוכחי הוא 6.

על סמך ייצוג זה ממשו את הפרוצדורות הבאות:

- number->sequence פרוצדורה המקבלת מספר ומייצרת סדרה שכוללת מספר זה בלבד.
- current-element פרוצדורה המקבלת סדרת מספרים המיוצגת על פי הגדרת הדקדוק למעלה, ומחזירה את האיבר הנוכחי בסדרה.
- move-to-left פרוצדורה המקבלת סדרת מספרים ומחזירה סדרת מספרים חדשה שבה האיבר הנוכחי הוא האיבר הקודם בסדרה שהתקבלה כפרמטר.
- move-to-right פרוצדורה המקבלת סדרת מספרים ומחזירה סדרת מספרים חדשה שבה האיבר הנוכחי הוא האיבר העוקב בסדרה שהתקבלה כפרמטר.
- insert-to-left פרוצדורה המקבלת מספר וסדרת מספרים ומחזירה סדרה חדשה שבה המספר שהתקבל כפרמטר התווסף לפני האיבר הנוכחי. לסדרה החדשה ימשיך להיות אותו איבר נוכחי.

- insert-to-right פרוצדורה המקבלת מספר וסדרת מספרים ומחזירה סדרה חדשה שבה המספר שהתקבל כפרמטר התווסף לאחר האיבר הנוכחי. לסדרה החדשה ימשיך להיות אותו איבר נוכחי.
- at-left-end? פרוצדורה המקבלת סדרת מספרים ומחזירה ערך בוליאני האם האיבר הנוכחי בסדרה הוא האיבר הראשון בסדרה.
- at-right-end? פרוצדורה המקבלת סדרת מספרים ומחזירה ערך בוליאני האם האיבר הנוכחי בסדרה הוא האיבר האחרון בסדרה.

להלן מספר דוגמאות שימחישו את השימוש בפרוצדורות אלה:

```
>(number->sequence 7)
(7 () ())
>(current-element `(6 (5 4 3 2 1) (7 8 9)))
6
>(move-to-left `(6 (5 4 3 2 1) (7 8 9)))
(5 (4 3 2 1) (6 7 8 9))
>(move-to-right `(6 (5 4 3 2 1) (7 8 9)))
(7 (6 5 4 3 2 1) (8 9))
>(insert-to-left 13 `(6 (5 4 3 2 1) (7 8 9)))
(6 (13 5 4 3 2 1) (7 8 9))
>(insert-to-right 13 `(6 (5 4 3 2 1) (7 8 9)))
(6 (5 4 3 2 1) (13 7 8 9))
```

- הפרוצדורות move-to-right ו-move-to-left אמורות להיכשל כאשר מנסים להפעילן על סדרה שבה האיבר הנוכחי הוא הימני ביותר או השמאלי ביותר בהתאמה.

א) כתבו אפיון לחתימות של הפעולות. סווגו את הפעולות השונות לסוגים (בנאים, מחלצים, מנבאים).

ב) כתבו אפיון אלגברי לכל אחת מהפעולות המגדיר את הסמנטיקה (המשמעות) שלהן.

הקפידו על הגדרה מלאה לכל פעולה.

ג) ממשו את טיפוס הנתונים המופשט על פי האפיון. ניתן לכתוב בנוסף פרוצדורות עזר במקרה הצורך.
אופן המימוש נתון לבחירתכם ויכול להיות ע"י רשימות או פרוצדורלי או ע"י שימוש ב-define-datatype

שאלה 2 (30 נקודות)

שאלה זו עוסקת בתחביר מוחשי (concrete syntax) ובתחביר מופשט (abstract syntax).

ביטוי חשבוני בכתיב prefix (כתיב פולני) הוא ביטוי שבו תחילה רושמים את הפעולה החשבונית שרוצים לבצע ולאחריה את האופרנדים הדרושים. בשאלה זו נעסוק רק בפעולת חיסור.

להלן נתון דקדוק המגדיר prefix-list שהיא רשימה שכוללת בתוכה ביטוי בכתיב prefix.

Prefix-list ::= (***Prefix-exp***)

Prefix-exp ::= ***Int***

::= - ***Prefix-exp*** ***Prefix-exp***

למשל,הביטוי בכתיב המוחשי (7 12 - 4 - 2 3 - -) הוא *Prefix-list* חוקי.

להלן נתון ייצוג של *Prefix*-exp באמצעות define-datatype:

```
(define-datatype  prefix-exp prefix-exp?
  (const-exp
    (num integer?))
  (diff-exp
    (operand1 prefix-exp?)
    (operand2 prefix-exp?)))
```

הביטוי הבא הוא דוגמא לביטוי הכתוב בכתיב מוחשי:

(7 12 - 4 - 2 3 - -)

להלן עץ תחביר מופשט (AST) עבור ביטוי זה:

```
(diff-exp
  (diff-exp
    (const-exp 3)
    (const-exp 2))
  (diff-exp
    (const-exp 4)
    (diff-exp
      (const-exp 12)
      (const-exp 7))))
```

כתבו פרוצדורה בשם parse-prefix המקבלת רשימה המייצגת *Prefix-list* חוקי (בתחביר מוחשי)

וממירה אותו לתחביר מופשט. (מדפיסה את ה-AST)

רמז: כתבו פרוצדורה שמקבלת רשימה ומחזירה *Prefix-exp* וגם את יתרת הרשימה שעדיין לא טופלה.

לפני פתרון השאלה רצוי לחזור על הדוגמא להמרת ביטוי מתחביר מוחשי לתחביר אבסטרקטי המופיעה בעמוד 53 בספר הלימוד.

שאלה 3 (20 נקודות)

בעמוד 40 בספר הלימוד נתון מימוש לטיפוס הנתונים **סביבה.** מימוש זה עושה שימוש בייצוג פרוצדורלי לייצוג הסביבה.
על סמך מימוש זה, כתבו פרוצדורה בשם ?has-binding המקבלת סביבה e ומשתנה v ובודקת האם ל- v ישנה כריכה תחת הסביבה e.
שימו לב, בשאלה זו הייצוג של סביבה הוא פרוצדורלי.

מטלת מנחה (ממ"ן) 13

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 3 בספר הלימוד, פרק 5 במדריך הלמידה, נספח B בספר הלימוד.

מספר השאלות: 2 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **28.4.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

לצורך פתרון מטלה זו עליכם ללמוד היטב את פרק 3 בספר הלימוד בליווי של פרק 5 במדריך הלמידה, וכן בנספח B בספר הלימוד.

שאלה 1 (50 נקודות)

הרחיבו את שפת LET (שפת "ויהי") עם ביטוי חדש בשם even-all .

להלן הדקדוק המגדיר את התחביר של even-all :

***Expression* ::= even-all [{ *Expression*}$^{*(,)}$]**

even-all-exp (exps)

ביטוי even-all פועל על רשימת ביטויים ובודק האם הערך של כל ביטוי ברשימה הוא זוגי, אם כן מוחזר true אחרת מוחזר false.

להלן דוגמא להמחשת השימוש בביטוי החדש :

```
let y= even-all  [4, –(7,1) , 90]
in
 if y then 100 else 200
```

בדוגמה זו, משתנה y יקבל ערך true מאחר וערכם של כל הביטויים ברשימה הוא זוגי, ולכן תוצאת כל ביטוי ה- let תהיה 100.

שאלה 2 (50 נקודות)

ברצוננו להרחיב את השפה עם ביטוי חדש find .

להלן הדקדוק המגדיר את התחביר של find :

Expression ::= **find** *Expression* <{ *Expression*}$^{*(\#)}$>

find-exp (val exps)

ביטוי find פועל על ביטוי val ורשימת ביטויים exps המופרדים בתו #. find מוצא את הביטוי הראשון ברשימה שערכו שווה לנתון val. במידה ונמצא ביטוי כזה find מחזירה את האינדקס בו נמצא הביטוי הראשון שערכו זהה ל-val (האינדקסים החל מ-1). במידה ואין ביטוי ברשימה שערכו זהה לערכו של val יוחזר 1- .

להלן דוגמאות להמחשת השימוש בביטוי החדש :

find 7 <4# -(8,5) # 90>

find 3 <4# -(8,5)#79#99>

בדוגמה הראשונה יוחזר 1- מאחר ו-7 אינו מופיע כלל ברשימת הביטויים.

בדוגמה השניה יוחזר 2 מאחר והביטוי השני ברשימה ערכו הוא 3.

מטלת מנחה (ממ"ן) 14

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 3 בספר הלימוד, פרק 5 במדריך הלמידה, נספח B בספר הלימוד.

מספר השאלות: 3 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **12.5.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

לצורך פתרון מטלה זו עליכם ללמוד היטב את פרק 3 בספר הלימוד בליווי של פרק 5 במדריך הלמידה, וכן בנספח B בספר הלימוד.

שאלה 1 (30 נקודות)

שאלה זו עוסקת בשפת "וישגור" (שפת PROC) המתוארת בספר הלימוד בפרק 3 בעמודים 74-81.

כזכור, שפת "וישגור" מאפשרת הגדרת פרוצדורות בעלות פרמטר יחיד.

להזכירכם, **משתנה חופשי** בפרוצדורה הוא משתנה המוזכר בגוף הפרוצדורה ואינו אחד מהפרמטרים של הפרוצדורה.

להלן דוגמא להבהרה,

```
let x=9
in
  let f= proc (y)  -(x,y)
  in
   (f 4)
```

בדוגמא זו, המשתנה x המוזכר בגוף הפרוצדורה f הוא משתנה חופשי, ואילו y הפרמטר של הפרוצדורה אינו משתנה חופשי.

האם שינוי שמו של הפרמטר של פרוצדורה (כלשהי) בשפת PROC לשם אחר, שעדיין שונה משמות המשתנים החופשיים בפרוצדורה, עלול לגרום לקבלת תוצאת ריצה שונה?

אם התשובה היא לא, הסבר מדוע?

אם התשובה היא כן, הבא דוגמא לתוכנית בשפת PROC הממחישה קבלת תוצאת ריצה שונה.

בתשובתך, התייחס למקרים הבאים:

א) מפרש בו הפרוצדורות מחושבות על פי כריכה סטטית (static binding)
ב) מפרש בו הפרוצדורות מחושבות על פי כריכה דינמית (dynamic binding)

שאלה 2 (30 נקודות)

א) בשפת "וישגור" (שפת PROC) הוגדר שלפרוצדורה יש רק ארגומנט יחיד. אך ניתן לעקוף מגבלה זו ולכתוב פרוצדורה עם מספר ארגומנטים ע"י שימוש בפרוצדורות שמחזירות בעצמן פרוצדורות.

נסתכל למשל על התוכנית הבאה בשפת "וישגור":

```
let f = proc (x) proc (y) …
in ((f 3) 4)
```

טכניקה זו נקראת Currying וכבר נתקלנו בה במסגרת השיעור הראשון בקורס.
כתבו בשפת "וישגור" באמצעות שימוש בטכניקה זו, פרוצדורה בעלת 2 ארגומנטים המחזירה את סכומם.
על מנת לחשב סכום של x+y בשפת "וישגור" ניתן לכתוב: -(x,-(0,y))

ב) להלן תוכנית בשפת "וישגור".

```
let makemult = proc (maker)
                proc (x)
                  if zero? (x)
                  then 0
                  else  -(((maker maker)  -(x,1)) , -4)
in  let times4 = proc (x) ((makemult makemult) x)
    in  (times4  3)
```

מהו הערך של תוכנית זו?

ג) כתבו בשפת "וישגור" פרוצדורה בשם factorial המחשבת עצרת של מספר.

לשם כך השתמשו בטכניקה מסעיף ב', וכן ב- Currying.

שאלה 3 (40 נקודות)

הרחיבו את שפת "וישגור" כך שפרוצדורות יוכלו לקבל מספר ארגומנטים, וכן קריאות לפרוצדורות יוכלו להיות עם מספר ארגומנטים. כפי שמוסבר בדקדוק הבא:

Expression ::= proc ({Identifier}$^{*(,)}$) Expression
Expression ::= (Expression {Expression}*)

מטלת מנחה (ממ"ן) 15

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 4 בספר הלימוד, פרק 6 במדריך הלמידה.

מספר השאלות: 2 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **26.5.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

לצורך פתרון מטלה זו עליכם ללמוד היטב את פרק 4 בספר הלימוד בליווי של פרק 6 במדריך הלמידה.

שאלה 1 (50 נקודות)

בספר הלימוד בעמודים 113-120 מתוארת שפת **"וירמוז"**(IMPLICIT-REFS).

ברצוננו לשנות את אופן ההגדרה וההפעלה של פרוצדורות בשפה זו. לשם כך נדרש לשנות את הביטויים proc-exp ו- call-exp.

להלן הדקדוקים המגדירים מחדש ביטויים אלה:

Expression ::= proc (*identifier* { , ref *identifier* }*) *Expression*

proc-exp (var refvars body)

Expression ::= call *Expression* (*Expression* {, *identifier* }*)

call-exp (rator rand args)

לביטוי **proc-exp** החדש המרכיבים הבאים:

- **var** – הפרמטר המועבר לפרוצדורה. בדיוק כמו בביטוי proc-exp.
- **refvars** – הם references למשתנים דרכם הפרוצדורה תחזיר את תשובותיה.
- **body** – הוא גוף הפרוצדורה.

לביטוי **call-exp** החדש המרכיבים הבאים:

- **rator** – ביטוי המתאר פרוצדורה
- **rand** – הארגומנט אותו רוצים להעביר לפרוצדורה.
- **args** – רשימת שמות של משתנים אליהם יוחזרו תשובות הפעלת הפרוצדורה.

להלן דוגמא :

```
let x = 9
  in let y=10
      in let p= proc (a,ref b,ref c)
                 begin
                  set b = -(a,1);
                  set c = -(a,2)
                 end
          in
            begin
               p(5,x,y);
               -(x,y)
            end
```

בדוגמה זו, מוגדרים 2 משתנים x=9, y=10 וכן פרוצדורה p המקבלת פרמטר אחד ועוד 2 פרמטרים המשמשים להחזרת שתי תשובות.
לפרוצדורה זו נשלח הערך 5 ושני המשתנים x,y (הרפרנסים שלהם), לאחר החזרה מהפעלת הפרוצדורה, ערכם של המשתנים הוא : x=4 , y=3 .
ולכן ערכה של התוכנית הוא (num-val 1)

שנו את שפת "וירמוז" והטמיעו בה שינויים אלה.

שאלה 2 (50 נקודות)

בספר הלימוד בעמודים 113-120 מתוארת שפת **"וירמוז"**(IMPLICIT-REFS).

ברצוננו להרחיב שפה זו עם ביטוי חדש שיאפשר לבצע לולאת for.
ערכו של ביטוי זה יהיה סכום ערכי הביטויים שבוצעו במהלך מחזורי הלולאה.

להלן הדקדוק המגדיר את הביטוי החדש שברצוננו להוסיף :

Expression* ::=** for (identifier= ***Expression ; ***Expression***; ***Expression***) ***Expression***

for-exp (tmpid initvalue bool inc body)

לביטוי **for-exp** המרכיבים הבאים:

- **tmpid –** משתנה זמני שמוכר רק במסגרת פעולת הלולאה. כלומר אחרי הלולאה חוזרים לעבוד עם הסביבה שהייתה טרם ביצוע לולאת ה-for.
- **initvalue –** ערך התחלתי שיינתן למשתנה הזמני לפני ביצוע מחזורי הלולאה.
- **bool –** ביטוי בוליאני שכל עוד ערכו **false** הלולאה תימשך (בניגוד ל-for המוכר לכם משפות אחרות שבו הלולאה נמשכת כל עוד התנאי נכון), כאן התנאי משמש כתנאי עצירה ברגע שערכו true.
- **inc –** ביטוי חשבוני שערכו יתווסף לערכו של המשתנה הזמני בכל סוף מחזור של הלולאה.

- **body** – ביטוי המשמש כגוף הלולאה.

שלבי פעולת הלולאה דומים ללולאת for המוכרת לכם משפות אחרות, בתחילה המשתנה הזמני מקבל את הערך ההתחלתי, לאחר מכן נבדק תנאי הלולאה, אם ערכו true הלולאה נעצרת, ואם ערכו false הלולאה נמשכת ועוברים לבצע את גוף הלולאה, לאחר ביצוע גוף הלולאה מוסיפים למשתנה הזמני את ערכו של הביטוי **inc**, ושוב חוזרים לבדוק את ערכו של הביטוי הבוליאני כמו קודם, וכן הלאה.
בסיום הלולאה, ערכו של ביטוי ה- for יהיה שווה לסכום ערכי הביטויים שבוצעו בגוף הלולאה במהלך מחזורי הלולאה.

להלן דוגמא :

```
for(x=10; zero?(x); -2)
    -(x,-100)
```

בדוגמה זו, הלולאה מתחילה עם ערך של 10 ובכל מחזור הוא קטן ב-2 עד להגעה לאפס, מה שיגרום לעצירת הלולאה. כלומר הלולאה רצה עם ערכי `x` הבאים : 10,8,6,4,2, בכל מחזור לולאה מבוצע גוף הלולאה, שהוא הביטוי `x+100`, ולכן ערכו של גוף הלולאה במחזורי הלולאה השונים הוא : 110,108,106,104,102 בהתאמה לערכי `x` השונים.
ערכו של ביטוי ה-`for` בסיום הלולאה יהיה סכום הערכים של גוף הלולאה בכל המחזורים שלה, דהיינו, 110+108+106+104+102 השווה ל- `(num-val 530)`.

הרחיבו את שפת "וירמוז" כך שתכלול תמיכה בביטוי החדש.

מטלת מנחה (ממ"ן) 16

הקורס: שפות תכנות (20905)

חומר הלימוד למטלה: פרק 7 בספר הלימוד, פרק 7 במדריך הלמידה.

מספר השאלות: 1 **משקל המטלה:** 5 נקודות

סמסטר: 2021ב **מועד הגשה:** **9.6.2021**

- שליחת המטלות תתבצע רק באמצעות מערכת המטלות המקוונת באתר הבית של הקורס.

לצורך פתרון מטלה זו עליכם ללמוד היטב את פרק 7בספר הלימוד בליווי הפרק המתאים במדריך הלמידה.

שאלה 1 (100 נקודות)

להלן נתונה תוכנית בשפת "ויקיש" (INFERRED):

((proc (x:?) x proc(y:?) y) 7)

ברצוננו לקבוע מהו טיפוס התוכנית (אם קיים).לשם כך, עליכם להשתמש באלגוריתם שנלמד בפרק 7 להקשת הטיפוס.

א. (40 נקודות)

הקצו לביטוי הנתון בשאלה ולתתי הביטויים שלו משתני-טיפוס (Type Variables) מתאימים, וחברו משוואות מתאימות לביטוי הנתון ולתתי הביטויים שלו.

ב. (60 נקודות)

פתרו את המשוואות שהרכבתם בסעיף א' תוך שימוש ב- substitution ו- unification בדומה למתואר בספר בעמודים 252-258. בסיום פתרון המשוואות רשמו מהו הטיפוס שאותו הקשתם עבור התוכנית הנתונה.