## שאלה 3

### A

נראה סדרת חישוב לפי call by value (כאשרa,b הם value)

test (and tru fls) a b =>

test false a b =>

b

### B.

**OR**

λb. λc. b true c

אם b הוא true נקבל:

(λt. λf. t) true c

לכן נחזיר true.  
זה נכון, כי לא משנה מה c בכל מקרה צריך להחזיר true

אם b הוא false נקבל:

(λt. λf. f) true c

לכן נחזיר את c.  
זה נכון, כי כעת אם c הוא true נקבל true, ואם c הוא false נקבל false – בדיוק כמו שOR היה מחזיר.

**NOT**

λb. b fls tru

אם b הוא true נקבל: false (כי הערך הראשון חוזר)

אם b הוא false נקבל: true (כי הערך השני חוזר)

**XOR**

λb. λc. not (and (b c true) (c b true))

נראה שכאשר הערכים של b ו-c שונים נקבל true, וכאשר עם זהים נקבל false:

אם b,c הם true נקבל:

λb. λc. not (and (b c true) (c b true)) => not (and c b) =>

not true =>

false

אם b,c הם false נקבל:

λb. λc. not (and (b c true) (c b true)) => not (and true true) =>

not true =>

false

אם b הוא true וc הוא false, נקבל:

λb. λc. not (and (b c true) (c b true)) => not (and c true) =>

not false =>

true

אם b הוא false וc הוא true, נקבל:

λb. λc. not (and (b c true) (c b true)) => not (and true b) =>

not false =>

true

## שאלה 4

### A

כן, התוצאה שווה ל-c1.

scc = λn. λs. λz. s (n s z)

c0 = λs. λz. z

**scc c0 =** (λn. λs. λz. s (n s z)) λs. λz. z => (E-AppAbs)

λs. λz. s ((λs. λz. z) s z) => (E-Abs)

λs. λz. s z = c1

### B

לא, התוצאה לא שווה ל-c1

(λn. λs. λz. s (n s z)) λs. λz. z => (E-AppAbs)

λs. λz. s ((λs. λz. z) s z) =>אין איך להמשיך, הגענו לערך

התוצאה שקולה ל-c1 במובן שהפעלה של הפונקציה הנ"ל על ערכים x,y כלשהם, תוביל לתוצאה זהה להפעלתם על הפונקציה c1:

λs. λz. s ((λs. λz. z) s z) x y => λz. x ((λs. λz. z) x z) y =>

x ((λs. λz. z) x y( => x ((λz. z) y) => x y .

c1 = (λs. λz. s z) x y => (λz. x z) y => x y

קיבלנו אותה תוצאה, לכן הפונקציות שקולות.

### C

**newScc = λn. λs. λz. c1 s (n s z)**

כאשר הערך של n הינו Church number – אותו נקדם באחד.

השתמשנו בהגדרה של plus (λm. λn. λs. λz. m s (n s z))- אשר מקבל שני מספרי church ומחבר אותם, וקבענו בו את m (המחובר הראשון) להיות c1, כלומר נחבר את מספר הקלט (n) תמיד עם c1 – קיבלנו scc.

### D

times = λm. λn. m (plus n) c 0

**power = λm. λn. n (times m) c1**

כאשר הפרמטר הראשון (m) הוא המעריך, והפרמטר השני (n) הוא החזקה.

בנינו זאת בצורה דומה לבניית פונקצית הכפל: פעולת חזקה היא הכפלה של המעריך בעצמו כמספר הפעמים של החזקה.

לדוגמא, נבחר מעריך c2 וחזקה c3, נקבל:

(λm. λn. n (times m) c1) c2 c3 >=

c3 (times c2) c1 =

(λs. λz. s s s z) (times c2) c1 =>

(times c2) (times c2) (times c2) c1 =>

(times c2) (times c2) c2 =>

(times c2) c4 =>

c8

### E

**isZero = λm. m (λb. fls) tru**

הגדרה: הפונקציה isZero מקבל מספר church ומעביר לו כפרמטר ראשון פונקציה שמקבלת פרמטר 1 ומחזירה false וכפרמטר שני את הערך tru.  
הסבר: המקרה היחידי בו אין שימוש בערך הראשון של הפונקציה(מספר church) ck הוא ב-c0, במקרה הזה פשוט חוזר הערך השני: לכן בפונקציה isZero נקבל tru.  
בכל מקרה אחר (ck>c0), נעשה גם שימוש בפרמטר הראשון שעברנו, ולכן תמיד נקבל fls. (נריץ את הפונקציה שמחזירה fls כמספר הפעמים של המספר church שהועבר כפרמטר – כל פעם על תוצאות הפונקציה fls בריצה הקודמת).