## שאלה 2

var x = 5;

function f(y) { return (x + y) - 2 };

function g(h) { var x = 7; return h(x) };

{ var x = 10; z = g(f) };

### A

הערך הוא 15

### B

x=10, y=7

### C

y הוא הערך שהועבר לפונקציה f כארגיומנט. הפונקציה נקראה מתוך הפונקציה g שהעבירה ל-f (באופן כללי ל-h, אבל בריצה שלנו הפונקציה f הועברה ל-g כארגיומנט) את x=7 שהוגדר לוקלית אצלה.

### D

x הוא הערך 10 שהוגדר בתוך הסוגריים המסולסלים האחרונים. הפונקציה f משתמשת בערך העדכני ביותר של x ה"קרוב ביותר" שהוגדר לפניה בעת ההרצה שלה (לא בעת ההגדרה שלה). מכיוון ש {} כלשעצמו בJAVA לא באמת פותחscope חדש, ה-x בשורה הראשונה והאחרונה שניהם מוגדרים באותו scope. (לעומת var x=7 שמוגדר לוקלית בתוך scope).

(function () {

var x = 5;

(function () {

function f(y) {return (x + y) - 2};

(function () {

function g(h) {var x = 7; return h(x)};

(function () {

var x = 10; z = g(f);

})();

})();

})();

})();

### E

הערך הוא 10.

### F

x=5, y =7

### G

**(כמו בדוגמא הקודמת)** y הוא הערך שהועבר לפונקציה f כארגיומנט. הפונקציה נקראה מתוך הפונקציה g שהעבירה ל-f (באופן כללי ל-h, אבל בריצה שלנו הפונקציה f הועברה ל-g כארגיומנט) את x=7 שהוגדר לוקלית אצלה.

### H

**בניגוד לדוגמא הקודמת**, כעת שורת ההרצה z = g(f), מוגדרת בתוך scope של פונקציה, ולכן כאשר נקרא לפונקציה f (דרך הפונקציה g) היא לא תכיר את המשתנים בscopes הפנימיים שקראו לה. ההגדרה הקרובה ביותר ביותר אליה כעת (שנמצא מעליה) היא var x = 5.