בס"ד

חלק 2 – תכנות python:

1. <u>קטע קוד 1:</u> ההדפסה תהיה [2,2,2] כי הפונקציה lambada: i לא שומרת את הערך של i בזמן יצירת הפונקציה, אלא רק מציינת שהיא תחזיר את הערך של המשתנה i (by reference) כשיקראו לה בעתיד.
בגלל שבסיום לולאת ה for, i = 3, כל הפונקציות שנשמרו ברשימה מחזירות 3.

```
functions = []
for i in range(3):
    functions.append(lambda: i)

print([f() for f in functions]) # first print
```

<u>קטע קוד 2:</u> ההדפסה תהיה [0,1,2] כי הפונקציה מוגדת כ lambada i: i, כלומר היא מקבלת פרמטר ברירת מחדל משלה בשם i (default parameter), שהערך שלו נקבע לפי הערך הנוכחי של i בלולאה באותו רגע - (by value). ככה בעצם כל פונקציה "זוכרת" את הערך העצמי הפרטי שלה, שנשמר בזיכרון כבר ביצירת הפונקציה

```
functions = []
for i in range(3):
    functions.append(lambda i=i: i)

print([f() for f in functions]) # second print
```

בעצם ההבדל בין שני קטעי הקוד הוא שבקטע הראשון הפונקציות שנוצרו שומרות הפניה למשתנה i, ולכן כולן מחזירות את הערך האחרון של i (אחרי שהלולאה הסתיימה)
לעומת הקטע השני שבה כל פונקציה שומרת עותק של הערך i במן היצירה שלה בזכות השימוש בפרמטר ברירת מחדל - ולכן הפונקציות מחזירות את הערכים שאליהם ציפינו.
** בשלב לולאת ה for הפעולה היחידה שמתבצעת היא **<u>יצירת פונקציה lambada והכנסה שלה לרשימת</u>**
<u>פונקציות functions!</u> ורק כשנקרא ל ()f בתוך ההדפסה – מתבצעת ההרצה האמיתי של הפונקציה, ואז רואים את הפלט של כל אחת מה lambada

2. ההדפסה תהיה [2,2,2] כמו בקוד הראשון מאותה הסיבה כמו מקודם, הפונקצייה lambada יוצרת פונקציה ששומרת הפניה למשתנה j מהסביבה החיצונית, ולא את הערך שלו בזמן יצירת הפונקציה. (ובסביבה החיצונית ערכו הוא 2 כערכו בסיום הלולאה.) ובזמן יצירת הפונקציה נשמרה רק הפניה למשתנה j ולא עותק של ערכו (ערך ממשי).

```
def make_lambdas():
    lambdas = []
    for i in range(3):
        j = i
        lambdas.append(lambda x: j + x)
    return lambdas

lambdas = make_lambdas()
for f in lambdas:
    print(f(0))
```

3. כדי לשנות את הקוד כך שיודפס כמו בקוד 2 נשתמש שוב בפרמטר ברירת המחדל בזמן יצירת הפונקציה ונכתוב את השורה הבאה מחדש, ככה:
lambdas.append(lambda x, j=j: j + x)
ככה בכל איטרציה של הלולאה, הערך הנוכחי של j יועתק לפרמטר של הפונקציה (by value) וכל פונקציה תזכור את הערך שהוקצה לה בזמן יצירתה כמו בקוד 2.