

טווח הכרה של משתתפים

בד"כ משתנים שאליהם התייחסנו, מוכרים בתוך
הפונקציה שבה הם מופיעים- משתנה מקומי. ישנן עוד
אפשרויות.

GLOBAL

אם יש לנו פונקציות מקוננות, הפונקציה פנימית תוכל להתייחס למשתנים שנימצאים בזו שמקיפה אותה. ניתן גם להגדיר משתנה 'גלובאלי' שיהיה מוכר בכל הפונקציות. משתנה כזה מוגדר בקובץ התוכנית מחוץ לכל הפונקציות. כדי לשנות אותו בתוך פונקציה כלשהי, צריכים להשתמש במילת המפתח global.

NONLOCAL

דוגמה למשתנה גלובאלי, אם מייבאים מודול כמו `math` ניתן להשתמש במשתנה בשם: `math.pi` שנימצא שם, בכל פונקציה שהיא בתוכנית.

אם יש פונקציה `b` המקוננת בתוך פונקציה `a`, והגדרנו משתנה בשם `x` (למשל `x=2`) בפונקציה החיצונית `a`, ועוד משתנה בשם `x` גם בפנימית `(b)`, הרי ש-`x` של הפנימית יאפיל' על ה-`x` של החיצונית. כדי להשתמש ב-`x` של החיצונית בפונקציה מקוננת, נגדיר אותו כ- `nonlocal`

סדר ההכרה בשמות משתנים הוא: **LEGB** (**L**ocal, **E**nclosing, **G**lobal, **B**uiltins) הנה מספר דוגמאות:

דוגמה

```
x=2
print('global x= ',x)
def main():
    x=4
    def inner():
        x=6
        print('inside inner x= ',x)
        # What happens if we comment out x=6 ?
        # What happens if we comment both x=6 and x=4 ?
    print('inside main x= ',x)
    inner()
main()
```

מה יודפס?

הדפסות

```
global x= 2  
inside main x= 4  
inside inner x= 6
```

ההשמה $x=6$ בתוך `inner()` מאפילה על $x=4$ שבתוך `main()`, והמשתנה הגלובאלי: $x=2$ גם מוסתר. אם נהפוך את השורה: $x=6$ להערה, ידפיס את שתי השורות הראשונות אותו דבר ובשורה האחרונה: `inside inner x=4` (ה- x של `main()`), ואם נהפוך להערה את $x=4$, ידפיס את הערך 2 בכל שלושת השורות.

דוגמה לשינוי משתנה גלובלי:

```
x=10
def main():
    x=5    # How can we change the global in main(),
           # so function1 can use the new value?
    print('inside main second x= ', x)
    global x
    x=4
    print('inside main x= ', x)
    function1()
def function1():
    # x is not assigned in function1(), so it must use global
    print('inside function1 x= ',x)
main()
```

מה יודפס?

הדפסות

inside main second x= 5
inside main x= 4
inside function1 x= 4

דוגמה לשימוש ב- nonlocal

```
x=10
def main():
    x=5 # local name for main
    print(' (first time) inside main x= ',x)
    def f1():
        nonlocal x # Looks in the enclosing def
        x=8 # changes the (x) local of main
        print('inside f1 x= ',x)
    f1()
    print(' (second time) inside main x= ',x) # local x of main changed by f1()
    f2() # f2() is not enclosed by main
def f2():
    print ('inside f2, x= ', x) # x here will be the global (no other x)
main()
```

מה יודפס?

הדפסות

(first time) inside main x= 5

inside f1 x= 8

(second time) inside main x= 8

inside f2, x= 10

ההדפסה השניה ב- main מראה לנו ש- f1() אכן שינתה את x שהוגדר ב- main() בגלל השימוש ב- nonlocal.