
לולאות ו-while



CYBER SCHOOL

מטרות



שיעור זה הינו מבוא לתכנות לולאות for-while בשפת פייתון.
במהלך השיעור, נזכיר דרכי ניהול כדי להדגיש את הפיתוח של קוד
מאורגן ויעיל.

לולאות for ו-while <





CYBER SCHOOL

לולאת ו-while

מה הן לולאות?



לולאות הן בלוקים של קוד שפועלות כל עוד תנאי מסוים מתקיים.

הן מאפשרות חזרה על לוגיקה של קוד מסוים, כל עוד התנאי נכון (true).



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help tests [C:\Users\Shayt\P\pycharm\projects\tests] - Mor loops.py - PyCharm
for loops.py
word = "test"
num = (1,2,3,4)

for i in word:
    print(i)

print()

for n in num:
    print(n)
```

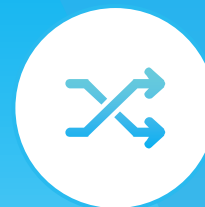
Run: for loops

```
t
e
s
t

1
2
3
4
```

Run Python Console Terminal Event Log
IDE and Plugin Updates: PyCharm is ready to update. (today 11:05) 13:5 CRLF UTF-8 4 spaces Python 3.7 (tests)

סוגי משתנים



מבצעים בלוקים של קוד אם התנאי נכון. <

לולאת **for** מספקת יכולות איטרציה. <

יכולה לבצע איטרציה על פני רשימות, טווחים מוגדרים ועוד. <



טווח לעומת רשימות בלולאות



איטרציה של רשימה מאפשרת הדפסת נתונים.
גם טווח קבוע, המוגדר על ידי `range()`, מסוגל לבצע איטרציה.
משתנה מוגדר בתוך הלולאה כפריט שעובר איטרציה.

```
Michigan Python - Range.py
Range.py
1 fruit = ["Watermelon", "Banana", "Apple", "Pineapple", "Strawberry"]
2
3 for i in fruit:
4     print(i, end=" ")
5
6 print("\n")
7
8 for i in range(0,5):
9     print(i, end=" ")
10
11 print("\n")
12
13 for i in range(0,len(fruit)):
14     print(i, end=" ")

Run: Range
Watermelon Banana Apple Pineapple Strawberry

0 1 2 3 4

0 1 2 3 4
Process finished with exit code 0
```



חישוב גודל לולאה

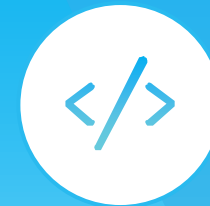
הפונקציה **len()** מחזירה את מספר הפריטים של אובייקט מסוים.
הפונקציה **range()** מקבלת integer ומחזירה את הטווח של האובייקט.

```
lesson.py
namelist = ["David", "John", "Cooper", "Nick"]
print(namelist, "The length is {}".format(len(namelist)))

colorlist = ["Blue", "Red", "Yellow"]

for i in range(8):
    print(i)
```

```
Run: lesson
['David', 'John', 'Cooper', 'Nick'] The length is 4
0
1
2
3
4
5
6
7
```



מעבדה - Range Loop



10 – 20 דקות

המשימה

יש לתרגל יצירת טווח עם קלטים שונים בלולאה.

השלבים

בקשת טווח.

חזרה על הטווח.

הצגת הערכים החוזרים.

כלים

פייתון 3

PyCharm

קבצים קשורים

מסמך מעבדה



CYBER SCHOOL

מעבדה – Loops in Nested Lists



15-25 דקות

המשימה

תרגול של הפעלת Nested Loops.

השלבים

פתיחת פרוייקט חדש ב-PyCharm ויצירת רשימה שנקראת "Classroom".

יצירת לולאה עבור כיתות (classrooms).

יצירת לולאה עבור התלמידים.

קבצים קשורים

מסמך מעבדה

כלים

פייתון 3
PyCharm



CYBER SCHOOL

יש לבצע בלוק קוד בזמן (while) שתנאי הוא נכון.
אם התנאי הוא False, הלולאה **while** לא תבוצע.
לולאות While אין יכולת איטרציה כמו לולאות for.

```
Michigan Python - While Loops.py
Michigan Python - Py-03 - While Loops.py
While Loops.py
counter = 0
while counter <= 6:
    counter += 1
    print(counter)

while counter <= 6
1
2
3
4
5
6
7

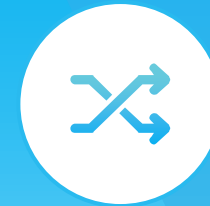
Process finished with exit code 0
Run | TODO | Problems | Terminal | Python Console | Event Lo
EP 8: W292 no newline at end of file
5:19 CRLF UTF-8 4 spaces Python 3.8 (Michigan Python)
```

לולאות While



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help tests [C:\Users\Shay\PycharmProjects\tests] - ...break_loops.py - PyCharm
tests break_loops.py
break_loops.py
counter = 0
while counter < 50:
    counter += 1
    if counter == 7:
        print("The counter reached the max number: {}".format(counter))
        break
    else:
        print("The current number is: {}".format(counter))
Run: break_loops
The current number is: 1
The current number is: 2
The current number is: 3
The current number is: 4
The current number is: 5
The current number is: 6
The counter reached the max number: 7
Process finished with exit code 0
IDE and Plugin Updates: PyCharm is ready to update. (today 11:05) 14:1 CRLF UTF-8 4 spaces Python 3.7 (tests) Event Log
```

פקודת Break



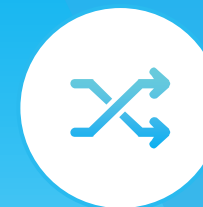
ניתן להפסיק/להפריע את פעולתן של שתי הלולאות, הן **for** והן **while**.

הפקודה **break** יוצאת מלולאה.

ניתן להציב אותה במיקום אסטרטגי כדי להשיג שליטה בזרימת הלולאה.



פקודת Continue



```
Michigan Python - Py-03 - Continue Command.py
Continue Command.py
1 counter = 0
2
3 while counter < 10:
4     counter += 1
5
6     if counter == 7:
7         print("Skipping one increment")
8         continue
9
10    print("Current number: {}".format(counter))
11
```

Run: Continue Command

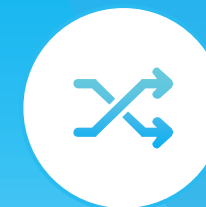
```
Current number: 1
Current number: 2
Current number: 3
Current number: 4
Current number: 5
Current number: 6
skipping one increment
Current number: 8
Current number: 9
Current number: 10
Process finished with exit code 0
```

ניתן להשתמש בהן בלולאות **for** והן בלולאות **while**.

הפקודה **continue** מדלגת על איטרציה.

מתנהגת כאילו הקוד הגיע לסוף הבלוק.

פקודת Pass



בהצהרות pass משתמשים כהרצות ריקות. ↩

הן שימושיות כאשר אנחנו לא רוצים להריץ שום קוד. ↩

ניתן להשתמש בפקודת pass כממלאת מקום (placeholder) עבור קוד שייכתב בעתיד. ↩

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help tests [C:\Users\Shayt\PycharmProjects\tests] - ..._break_loops.py - PyCharm
tests continue.py
continue.py
counter = 0
while counter < 50:
    counter += 1
    if counter == 7:
        pass
    else:
        print("The current number is: {}".format(counter))

Run: continue
The current number is: 1
The current number is: 2
The current number is: 3
The current number is: 4
The current number is: 5
The current number is: 6
The current number is: 8
The current number is: 9
The current number is: 10
The current number is: 11
```





שאלות?