**פייתון - הרצאה מספר 4**

**הפונקציה range**

פונקציה זו מייצרת רצף של מספרים שלמים.

הפונקציה range היא אחת מהפונקציות השימושיות והנפוצות ביותר בפייתון. range מייצרת טווח של מספרים שלמים.

**היא מקבלת בתור פרמטרים ערכים שלמים: התחלה, סוף וקפיצה, ויוצרת סדרת מספרים.**

range(start, **stop**, step)

**start** – פרמטר אופציונלי, מספר שלם המציין את תחילת הרצף. ערך ברירת המחדל - 0 .

**stop** – פרמטר הכרחי. מספר שלם המציין את סוף הרצף (לא נכלל ברצף).

**step** - פרמטר אופציונלי, מספר שלם המציין את ההפרש בין המספרים ברצף, ערך ברירת מחדל - 1.

**דוגמאות:**

print(range(10))

print (list(range(10)))

יודפס:

range(0, 10)

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

הוראה זו מדגימה את ערכי ברירת המחדל של :range  
כאשר היא מקבלת רק פרמטר אחד, היא מניחה שיש צורך להתחיל מ- 0 ולהתקדם בקפיצות של 1 לכן נקבל את הערכים 0 עד 9 בקפיצות של 1.

**כדי להציג את סדרת הערכים שנוצרה ע"י range נמיר אותה לרשימה ע"י הפונקציה .list**

print (list(range(5,12)))

הוראה זו יוצרת רשימת מספרים שמתחילה ב- 5, עוצרת לפני 12, בקפיצות של 1.

[5, 6, 7, 8, 9, 10, 11]

במקרה כזה יש להקפיד שהפרמטר השני גדול מהראשון, אחרת נקבל רשימה ריקה.

print (list(range(50,12)))

[ ]

* נוכל לייצר גם טווח מספרים שלילי.

print (list(range(-10, -3)))

הוראה זו יוצרת טווח שמתחיל ב- 10- ומגיע עד ) -3 לא כולל ) בקפיצות של 1.

[-10, -9, -8, -7, -6, -5, -4]

* כל הטווחים שראינו עד כה מתקדמים כברירת מחדל בקפיצות של 1. אפשר לשנות זאת בעזרת פרמטר שלישי.

print (list(range(20,55,5)))

הוראה זו יוצרת טווח שמתחיל ב- 20 ומגיע עד ) 55 לא כולל ) בקפיצות של 5.

[20, 25, 30, 35, 40, 45, 50]

* range יכולה ליצור רצפים של מספרים בלבד, כל טיפוס אחר שננסה לתת כפרמטר יוביל לשגיאה.

print(range('b','h')) **// not ok**

**מבני בקרה – לולאות**

לעיתים במהלך פתרון בעיות יש צורך לחזור ולבצע סדרת הוראות מספר פעמים ברצף. סדרת הוראות החוזרת על עצמה מספר פעמים נקראת לולאה (loop). שני מבני בקרה מרכזיים לביצוע חוזר הם: לולאת **for** ולולאת **while**.

**לולאת for**

לולאת for מתבצעת מספר מוגדר של פעמים. היא שימושית במיוחד כאשר יש לנו אוסף של איברים שאנחנו רוצים לבצע עליהם את אותה פעולה/ סדרת פעולות. לולאות forבפייתון נכתבות בצורה מעט שונה משפות תכנות אחרות. אנחנו לא מגדירים בפייתון את מספר הריצות בצורה ישירה. במקום זאת אנו מספקים ללולאה רצף כלשהו והלולאה מתבצעת עבור כל אחד מהערכים ברצף.

**פורמט ההוראה:**

for <איטרטור – משתנה הלולאה > in < רצף איברים > :

<הוראה / הוראות לביצוע >

הלולאה מקבלת רצף של איברים, לדוגמה: מספרים מ – 1 עד 10/ ארבעה שמות של ילדים או כל אוסף אחר. בכל איטרציה ( מעבר על הלולאה ) אחד האיברים מרצף האיברים נטען לתוך משתנה שעליו רצה הלולאה (איטרטור). ההוראה/ ההוראות בלולאה מתבצעות. לאחר סיום האיטרציה, נטען האיבר הבא מרצף האיברים וכך הלאה עד סיום כל האיברים ברצף.

**דוגמה:**

for i in [0,1,2]:

print(i\*2)

לולאת forמתחילה במילה , for לאחר מכן יבוא שם של משתנה כלשהו שעליו רצה הלולאה ונקרא "איטרטור" ((iterator, אחר כך תופיע המילה ,inולאחר מכן רצף של איברים ונקודותיים. בדוגמה הבאה אנחנו שמים כמה מספרים בתוך סוגריים מרובעים מה שאומר שהם הופכים ל"רשימה", טיפוס של משתנה שנלמד עליו בהמשך.

ההוראה/ההוראות שיש לבצע בלולאה יוזזו באותו אופן כמו בבלוק הוראות של הוראת if.

פלט ההרצה יהיה:

0

2

4

**דוגמה נוספת שרצה על אוסף של מחרוזות:**

for i in ['Ben Gurion', 'Sharet', 'Begin', 'Rabin']:

print(i + ' was the prime minister of Israel')

פלט ההרצה יהיה:

Ben Gurion was the prime minister of Israel

Sharet was the prime minister of Israel

Begin was the prime minister of Israel

Rabin was the prime minister of Israel

עבור הקוד הבא:

letter\_list = ['a','b','c']

for letter in letter\_list:

print(letter)

יודפס:

a

b

c

מה אם נרצה לעשות לולאת forשעוברת על סדרה של מספרים רבים? די לא נוח לכתוב את כל המספרים אחד אחרי השני.

= > נעזר בפונקציה range.

**תרגיל:**

ניצור לולאת for שמדפיסה את כל המספרים מ- 1 עד 40 (כולל) זה לצד זה, מופרדים ע"י פסיקים, פרט לאיבר האחרון, אחריו לא יופיע פסיק.

for i in range(1,41):

if (i!=40):

print(i, end=",")

else:

print(i)

**תרגיל:**

כתבו תכנית המיישמת את המשחק "7 בום" בתחום המספרים 1 עד 99(כולל). אם

המספר מתחלק ב- 7 או מופיעה בו הספרה 7 תודפס המילה boom"", אחרת יודפס

המספר עצמו.

**פתרון:**

for i in range(1,100):

units = i % 10

tens = i // 10

if units==7 or tens==7 or i%7==0:

print("boom")

else:

print(i)

**שימוש ב- pass**

אם נרצה שהלולאה שכתבנו לא תבצע כלום נעשה שימשו בהוראה pass.

למה שנרצה לכתוב קוד שלא מבצע כלום? הדבר שימושי בתהליך פיתוח קוד. אנחנו כותבים את שלד התוכנית, שמכיל לולאות ופונקציות, אבל משאירים אותן ריקות. כך אנחנו מסיימים במהירות את השלד ויכולים לבדוק שהתכנון של הקוד שלנו הוא נכון, לפני שאנחנו מתחילים להתעסק עם המימוש עצמו. זה די שימושי כאשר כותבים תכניות מורכבות.

לדוגמה:

for i in range(7):

pass

לולאה זו תרוץ 7 פעמים ובכל פעם לא תעשה דבר. כרגע זה אולי לא נראה שימושי במיוחד, אבל כאשר נכתוב תכניות עם פונקציות, זה יהיה די מועיל בשלב תכנון הקוד.

**לולאת while**

לולאת while היא לולאה תלוית אירוע המתבצעת כל עוד תנאי מסוים מתקיים.

במקרים בהם לא ידוע מראש מספר הפעמים שיש לחזור על ביצוע גוף הלולאה, והלולאה מתבצעת בהתאם לקיומו של תנאי בוליאני, מקובל להשתמש בלולאה מותנית.

**יש לוודא שהלולאה איננה אינסופית.**

**לולאת while** מתבצעת כל עוד ערכו של הביטוי הבוליאני "אמת". יכול להיות שלולאת while לא תתבצע כלל.

**פורמט ההוראה:**

: **ביטוי בוליאני**while

**הוראה/ הוראות לביצוע**

**הערות:**

**ביטוי בוליאני** - יכול להיות פשוט או מורכב והוא תנאי הכניסה ללולאה. כלומר, אם התנאי מתקיים ההוראה/ההוראות בבלוק שאחריו יתבצעו.

**מה יבצע הקוד הבא:**

number = 1

while number<11:

print(number)

**הקוד ידפיס אינסוף שורות של המספר 1, כיון שהתנאי מתקיים תמיד.**

**מה צריך לעשות כדי שהלולאה תדפיס את המספרים 1- 10 (כולל) ואז תיעצר?**

יש להוסיף הוראה אחת לבלוק ההוראות של while:

number = 1

while number < 11:

print(number)

number+=1

**דוגמה:**

**מה יבצע הקוד הבא?**

age = 17

while age < 18:

age += 1

print('Not yet 18')

המשפט ""Not yet 18 יודפס פעם אחת. באיטרציה השנייה של הלולאה התנאי כבר לא יתקיים ולא תהיה כניסה לתוך הבלוק.

**דוגמה:**

שני תלמידים המשחקים זה נגד זה, מותחים בתחילת המשחק קו באורך N ס"מ (N מספר טבעי ו- גדול מ-1). השחקנים מחליפים תורים לסירוגין. כל שחקן בתורו מקצר את הקו לחצי מאורכו. השחקן אשר מקצר בתורו את הקו לאורך של פחות מ-1 ס"מ מנצח במשחק. למשל, אם אורכו התחילי של הקן הוא 8 ס"מ, אזי השחקן הראשון יקצר את הקו ל 4, השחקן השני יקצר את הקו ל – 2, השחקן הראשון יקצר את הקו ל- 1 והשחקן השני יקצר את הקו ל- 0.5 וינצח במשחק. כתבו תכנית אשר קולטת את אורכו התחילי של הקו ומדפיסה מי מנצח במשחק (הראשון מנצח או השני). **הערה:** יש לפתור באמצעות לולאה

**פתרון:**

len= int(input("Please enter the lines Length "))

count= 0

while len>=1:

len/=2

count+=1

if(count %2 ==0):

print("The winner is: Player no. 2")

else:

print("The winner is: Player no. 1")

**לולאות מקוננות**

**לולאה** יכולה להיות **מקוננת**, ז"א גוף לולאה יכול להכיל לולאה נוספת. עבור כל איטרציה בלולאה החיצונית, הלולאה הפנימית מופעלת מתחילתה ועד סופה.

**דוגמה: מה ידפיס הקוד הבא?**

numbers\_list=[1,2,3,4,5,8,5,4,3,2,1]

for i in numbers\_list:

print(i,' - ',end='')

for j in range(i):

print('\*',end='')

print()

**הקוד ידפיס:**

1 - \*

2 - \* \*

3 - \* \* \*

4 - \* \* \* \*

5 - \* \* \* \* \*

8 - \* \* \* \* \* \* \* \*

5 - \* \* \* \* \*

4 - \* \* \* \*

3 - \* \* \*

2 - \* \*

1 - \*

**שימוש ב- break ו- continue:**

**פקודת break** אקטואלית עבור כל הלולאות.

,breakפירושו הפסקה מוחלטת של הלולאה (בה הוא נמצא) ומעבר לביצוע ההוראה העוקבת ללולאה.

**לדוגמה:**

מה יודפס כתוצאה מהרצת הקוד הבא:

N = 5

factorial = 1

for i in range(1, N+1):

if (i==3):

break

factorial = factorial \* i

print("factorial=", factorial)

עבור N=5 יתבצע: 2\*1 והפלט יהיה 2 factorial =

מה יבצע הקוד הבא:

while True:

print ("Hi")

זוהי לולאה אינסופית. הריצה שלה לעולם לא תסתיים.

מה יבצע הקוד הבא:

while True:

print ("Hi")

break

print ("Bye")

שורה 2 תדפיס .Hi בשורה 3אנחנו מורים למחשב לצאת מהלולאה, לכן שורה 4 לעולם לא תרוץ.

וכעת, מה יבצע הקוד הבא:

while True:

print ("Hi")

while True:

print ("Bye")

break

רצף אינסופי של " "Hiואז " ."Byeה – break שבשורה 5יגרום לתוכנית לצאת מהwhile - שבשורה ,3אך לא מ ה- while שבשורה .1 לכן הלולאה שבשורה 1היא עדיין לולאה אינסופית, ואילו הלולאה שבשורה 3נקטעת בכל פעם מחדש במהלך הריצה.

**פקודת continue** אקטואלית עבור כל הלולאות.

מסיימת איטרציה נוכחית בלבד ומנסה לבצע את האיטרציה הבאה.

**לדוגמה:**

מה יודפס כתוצאה מהרצת הקוד הבא:

N = 5

factorial = 1

for i in range(1, N+1):

if (i==3):

continue

factorial = factorial \* i

print("factorial=", factorial)

עבור N=5 יתבצע: 5\* 4\* 2\* 1והפלט יהיה 40factorial =

**דוגמה נוספת:**

מה יודפס כתוצאה מהרצת הקוד הבא:

number = 0

while number <= 20:

number += 1

if number % 5 != 0:

continue

print(number)

הקוד ידפיס את המספרים בתחום 0 - 20 (כולל) שמתחלקים ב 5

5

10

15

20

**תרגיל:** כתבו תכנית הקולטת 5 מספרים ומדפיסה כמה מהמספרים הנ"ל הם מספרים ראשוניים. (מספר ראשוני הינו מספר טבעי - שלם וחיובי שמתחלק רק ב- 1 ובעצמו. 1 לא נחשב למספר ראשוני, ואילו 2 נחשב למספר הראשוני הקטן ביותר).

import math

N = 5

**# מונה מספרים ראשוניים**

count2 = 0

for j in range(N):

**# num מונה את מספר המחלקים של**

count1 = 0

num = int(input("please enter a natural number "))

if num == 2:

count2 += 1

elif num!=1:

**#sqrt - returns a "float" number => floor**

for i in range(2, math.floor(math.sqrt(num))+1):

if (num % i == 0):

count1 += 1

if count1 == 0:

count2 += 1

print(count2,"primary numbers where found ")