**פייתון - הרצאה מספר 1**

פייתון Python) ) היא שפת תכנות מונחית עצמים נפוצה מאוד בעלת שימושים רבים. הרעיון המקורי של פייתון, שפותחה בראשית שנות ה-90, הוא לפשט קונספטים מורכבים משפות תכנות אחרות, להפוך את הסינטקס (תחביר השפה) לקריא ואינטואיטיבי יותר ולאפשר למפתחים ליצור תכניות מורכבות בזמן פיתוח קצר יחסית תוך שימוש בפחות שורות קוד. השפה פותחה ע"י מפתח הולנדי בשם גווידו ואן רוסום ובמרוצת השנים צברה קהילת משתמשים גדולה ונלהבת.

פייתון הינה שפת סקריפטים, ז"א כל פקודה שאנחנו כותבים בפייתון מתורגמת לשפת מכונה רק בזמן הריצה. במידה ויש שגיאה בקוד, ההוראות בשפה ירוצו עד שארעה שגיאה, בניגוד לשפה כמו C# למשל, שלא תרוץ כלל עד שלא נתקן את כל שגיאות הסינטקס.

**הוראת קלט**

הוראת קלט מאפשרת לקבל ערכים למשתנים בזמן הרצה.

**פורמט ההוראה:**

= input (' Please enter… ');שם של משתנה

**דוגמאות:**

fname = input ('please enter your fist name');

salary =input ('please enter salary');

**הערות:**

* כאשר נשתמש בהוראת קלט, נוסיף לפניה הוראת פלט מקדימה המציגה הודעה למשתמש לגבי הנתון שיש להדפיס.

למשל: "הקלד שם משתמש"

"הקלד מספר זהות (בספרות בלבד)" ...

**הוראת פלט**

הוראת פלט מאפשרת להציג למשתמש מידע הקשור בביצוע התכנית.

**פורמט ההוראה:**

print (רשימת פלט )

**רשימת פלט יכולה להכיל:** קבועים, משתנים, הודעות ושילובים שונים שלהם.

כל הוראת print תיצור שורת פלט חדשה.

**דוגמאות:**

print( 'Good Day')

הדפסת הודעה, תחומה בין גרשיים.

הפלט שיוצג על המסך: Good Day

ניתן לשרשר מחרוזות בהוראת הפלט.

print( 'Good ' + 'Day' )

הפלט שיוצג על המסך: Good Day

או להפריד ביניהם ע"י פסיקים

print( 'Good' , 'Day' )

הפלט שיוצג על המסך: Good Day

פייתון לא יכולה לשרשר ערכים מטיפוסים שונים (כמו מחרוזת ומספר).

**print('Have a Good Day for ' + 24 + ' hours a day ') // not ok**

כדי להתגבר על הבעיה אנחנו יכולים להשתמש בפונקציית המרה או להכניס את המספר בתור מחרוזת טקסט ע"י הקפה שלו בגרש / בגרשיים:

print('Have a Good Day for ' + str(24) + ' hours a day ')

,או

print('Good Day all day for ' + '24' + ' hours ')

אפשרות נוחה יותר, היא הדפסת ערכים נפרדים באמצעות פסיקים (במקום שרשור שלהם באמצעות "+" ). בצורה זו טיפוסי הערכים לא משנים, הם מודפסים "זה לצד זה" אבל התוצאה נראית זהה.

print('Good Day all day for ' , 24 , ' hours ')

\n - יורד שורה

הפרמטר end מגדיר לפונקציה print איזה תו אנחנו רוצים שיופיע בסוף ההדפסה. כברירת מחדל התו - '\n' - ירידת שורה, מופיע כתו בסוף הפונקציה ) printאנו לא רואים זאת ). זאת הסיבה שכל הדפסה נוספת שתתבצע תמיד תהיה בשורה חדשה. הצבת ערך שונה בפרמטר end יבטל את ירידת השורה של ההדפסה הבאה ובכך יגרום לשתי השורות "להיצמד" יחד.

דוגמה:

print ("My name's Bond", end="->")

print("James Bond")

יודפס: My name's Bond-> James Bond

אם מעוניינים רק בביטול ירידת שורה ללא הוספת טקסט כלשהו, אפשר להשאיר את הגרשיים ריקים.

print ("My name's Bond ", end=" ")

print("James Bond")

יודפס:

My name's Bond James Bond

הדפסת טקסט בשילוב עם ערכי משתנים:

דוגמה:

a=input("number of apples to eat every day ")

print("eat",a,"apples every day")

על המסך יוצג:

number of apples to eat every day 5

eat 5 apples every day

שימו לב, פסיק "מכניס" רווח להדפסה

**הערות - תיעוד בקוד**

הערות מיועדות לתיאור: משתנים, מטרת התכנית ומרכיביה השונים.

המהדר מתעלם מההערות.

**הערה הנכתבת בשורה אחת #**

**""" הערה הנכתבת**

**""" במספר שורות**

**מה כדאי לתעד:** משתנים, פונקציות, קטעי קוד מעניינים...

**משתנים**

בפייתון כל משתנה הוא אובייקט. משתנים לא "מכילים" ערכים בעצמם אלא מצביעים לערכים שמאוחסנים בזיכרון. זאת הסיבה שכל כך קל להגדיר משתנים בפייתון. אין צורך להגדיר סוג כי למשתנה אין סוג – הוא פשוט מצביע לערך מסוג מסוים. לכן הצבה מחודשת של ערך בתוך משתנה היא לא יותר מאשר הזזת ההצבעה לערך אחר. בגלל העיקרון הזה כל משתנה יכול להכיל כל סוג וגם להחליף סוגי מידע שונים ללא שום בעיה.

**שם משתנה (מזהה) identifier**

* שם חייב להתחיל באות באנגלית, ויכול להכיל אותיות, ספרות ומקף תחתון.
* Case-Sensitive is Python. השפה מבדילה בין אותיות קטנות וגדולות לכן יש להקפיד ולהשתמש במדויק באותו שם משתנה במהלך כתיבת הקוד. לדוגמה: num שונה מ- Num.
* שם לא יכול להיות מילה שמורה.

בפייתון 28 מילים שמורות

and       continue  else      for       import    not       rais   
assert    def       except    from      in        or        return  
break     del       exec      global    is        pass      try   
class     elif      finally   if        lambda    print     while

* יש לתת למשתנים שמות משמעותיים שיעידו על תפקידם בתוכנית.

דוגמאות: average\_of\_marks sum, number,.

**הערה: אם מנסים להשתמש בשם לא חוקי עבור משתנה, מתקבלת הודעת שגיאה.**

**טיפוסי משתנים בסיסיים שכיחים**

כדי להשתמש במשתנה בפייתון לא צריך להגדיר מאיזה סוג הוא יהיה, פשוט צריך לתת לו שם חוקי ולהציב בו ערך. סוג הערך שנציב במשתנה יקבע את טיבו.

כמו ב – java script גם כאן ניתן לעשות הצבה מחודשת למשתנה ולשנות את סוג הערך שמוצב בו ( לא רצוי לעשות זאת – עלול לגרום לבלבול).

**טיפוסי משתנים בסיסיים בפייתון :**

מספר שלם integer : 1, 13 וכו'

מספר עשרוני floating point: 0.5 , 3.14 וכו' **\*\***

מחרוזת "hello world " : string

משתנה בוליאני : True, False ( למשתנים בוליאניים יש גם ערך מספרי True=1, False=0)

**\*\*** - במספרים מסוג float קיים אי דיוק זעיר.

לדוגמה : print(0.1+0.1+0.1) יודפס: 0.30000000000000004

**ניתן לבצע בשפה את הפעולות הבאות:**

name="hello "

print(name\*2)

הוראות אלו יציגו את המחרוזת במשתנה name פעמיים: hello hello

* הצבה מרובה בשורה אחת

a,b,c = 10,20,30

print (a,b,c)

ההוראה תציג : 30 20 10

* **לא ניתן לחבר מחרוזת עם מספר, למשל:**

**word="3"**

**number=5**

**print(word+ number)**

**תתקבל שגיאה כתוצאה מהרצה זו.**

ניתן להתגבר על כך ע"י המרת המספר למחרוזת או המרת המחרוזת למספר:

word="3"

number=5

print( **int(**word**)**+ number )

יתקבל הפלט: 8

\* המרה למחרוזת מתבצעת ע"י הפונקציה str

word="3"

number=5

print (word+**str(**number**)**)

יתקבל הפלט : 35

\* **המרה למספר ממשי:** ( ערך להמרה **float(**

**בדיקת סוג המשתנה על ידי type**

לבדיקת סוג המשתנה נעשה שימוש בהוראה type

דוגמה:

word="3"

print(type(word))

יוצג: < 'class 'str>

number= 5.7

print(type(number))

יוצג: < 'class 'float>

**a=5==7**

**print(type(a))**

**print(a)**

**יוצג:** < 'class 'bool>

False

**אופרטורים מיוחדים**

price = price + 5; זהה ל :price+= 5;

price = price - 8; זהה ל : price -= 8;

price = price \* 1.18; זהה ל : price \*= 1.18;

price = price / 5; זהה ל : price /= 5;

**פעולות חשבוניות:**

|  |  |
| --- | --- |
| **סימן** | **פעולה** |
| + | חיבור |
| - | חיסור |
| \* | כפל |
| / | חילוק  דוגמה: 10/2 יוחזר הערך 5.0  חילוק שני מספרים תמיד יחזיר מספר עשרוני |
| // | חילוק מספרים שלמים לקבלת ערך שלםדוגמה: 17 // 5 יוחזר הערך 3 |
| \*\* | חזקה  דוגמה: 4 \*\* 2 יוחזר הערך 16 |
| % | שארית כתוצאה מחלוקה  דוגמה: 11%3 יוחזר הערך 2 |

כל תוצאה של פעולה חשבונית בין מספר מסוג int ומספר מסוג float תמיד תשמר בתור float.

**תווים**

בשפת פייתון אין טיפוס נתונים שונה לייצוג תווים. תוו בודד הינה מחרוזת באורך 1.

תווים מיוצגים ע"י קוד ASCIIכאשר לכל תו יש מספר שמייצג אותו בקוד.

הפונקציה chr( ) - מקבלת מספר ומחזירה את התו שמספר זה מייצג בקוד ASCII.

הפונקציה ord ( ) – מקבלת תו ומחזירה את המספר בקוד ASCII שמייצג אותה.

דוגמאות:

print("chr(48)=",chr(48))

print("ord(A)=",ord('A'))

יודפס:

chr(48)= 0

ord(A)= 65

**ביטויים בוליאניים**

ביטויים שיכולים להיות נכונים או לא נכונים וחישוב שיתבצע עליהם יחזיר את הערך True במידה והביטוי שרשמנו הוא נכון, והערך False יחזור בכל מקרה אחר.

print(7>3)

print(3\*5==15)

print("z" in "Go ahead, make my day.")

print("Go ahead" in "Go ahead, make my day.")

יודפס:

True

True

False

True