בס"ד

מבוא לפייתון / סיכום 3 + תרגול

סיכום

פונקציות:

הגדרת פונקציה – :def func_name(variables)

כל פונקציה מקבלת בדיוק n משתנים שהוגדרו מראש.

args* – הופך את כל הפרמטרים שאין להם התאמה ל-tuple

kwargs** - הופך את כל הצמדים של מפתח-ערך שאין להם התאמה ל-dictionary.

החזרת ערך מהפונקציה ע"י return. לא חובה. ניתן להחזיר כל ערך כולל sequences ורקורסיה.

הפרמטרים עוברים by reference ולא by value, כלומר, מועבר מצביע למשתנה.

אם משתנה מוגדר בתוך פונקציה, כל ההפניות אליו הן מקומיות, כולל הפניות לפני שהמשתנה הוגדר!

משתנה גלובלי – global

משתנה לא מקומי ביחס לפונקציה פנימית וחיצונית – nonlocal

פונקציה אנונימית – lambda

map – פונקציה המקבלת פונקציה וסדרה ומחשבת פלט עבור סדרת הקלטים.

filter – פונקציה המקבלת פונקציה וסדרה ומוציאה רק פלטים שהם True.

תכונות של פונקציות:

קבלת תכונות של פונקציה – dir(func_name)

__doc__ - תכונה המביאה את התיעוד של הפונקציה.

__call__ - תכונה המפעילה את הפונקציה.

__defaults__ - None או Tuple שמכילים ערכי ברירת מחדל של הפרמטרים.

__code__ - תכונה שבה כל נמצאות התכונות החשובות של הפונקציה.

<u>מחלקות:</u>

הגדרת מחלקה – :class ClassName

כל מחלקה יורשת ממחלקת הבסיס object. מופיע אחרי שם המחלקה בסוגריים.

בפייתון, משמעות הירושה היא אחת: אם מתודה לא נמצאת במחלקה, נחפש אותה במחלקת האב.

help(ClassName) - מידע על כל הפונקציות של המחלקה ClassName בנוסף לשורת התיעוד.

__init__ - פונקציה המשמשת כבנאי המחלקה.

__bases__ - מחזירה tuple המכיל את מחלקת הבסיס (מחלקת ה"אב").

בפייתון הכל public.

מוסכמה: משתנה ששמו מתחיל בקו תחתון (_) נחשב כ-nonpublic.

שימושי כאשר נבנה תתי מחלקות ונרצה לכתוב פונקציות בעלות שם זהה מבלי לדרוס אחת את השניה.

אפשר להגדיר תכונה של מחלקה ע"י הגדרתה ברמת המחלקה. במקרה כזה, **אין** self. כל מה שנדרש זה השמה למשתנה והוא יוגדר במחלקה.

בדיקה אם מחלקה A היא תת-מחלקה של המחלקה B ע"י הפונקציה issubclass(A, B).

בדיקה אם a הוא מופע של המחלקה A ע"י הפונקציה isinstance(a, A).

<u>:self</u>

כל השיטות במחלקה תמיד מקבלות כפרמטר ראשון פרמטר בשם self, שהוא האובייקט החדש שנוצר (instance).

משתנה ולא מילה שמורה!

זהו המופע הנוכחי. בדומה ל-this בשפות אחרות.

לא לשכוח להגדיר אותו בכל בנאי –זוהי מוסכמה חזקה.

<u>ירושה:</u>

ע"מ להימנע מבעיות בירושה בפייתון:

אפשרות אחת: להוסיף בשורה הראשונה של פונקצית האתחול __init__ את השורה fatherClass.__init__(self,...).

אפשרות טובה יותר: super(ClassName, self).__init__(parameters of father's class).

בעת ירושה מיותר ממחלקה אחת, אם יש משתנה המופיע בכמה מחלקות עם אותו שם, הוא יוכר במחלקת הבן בתור משתנה של המחלקה הראשונה ברשימה המחלקות!

<u>מודול:</u>

הקובץ המכיל את המחלקה הוא בעצם מודול שאנחנו יצרנו.

קריאה למודול ע"י from <module> import <class>

לדוגמא, אם בנינו בקובץ F.py מחלקה Family המגדירה משפחה, נוכל בקובץ ההרצה example.py להוסיף את שורה from F import Family ובכך לייבא את המחלקה Family מהמודול F!

<u>חריגות:</u>

טיפול ע"י try-except.

הדפסת שגיאה "מותאמת אישית" ע"י raise.

לאחר קטע ה-except ניתן להוסיף משפט else שיתבצע אם ה-except לא התבצע.

ניתן גם להוסיף משפט finally שיתבצע בכל מקרה.

תרגילים

1. כתבו פונקציה שמשתמשת בחיפוש בינארי (הסבר בהמשך המסמך) למציאת פריט ברשימה ממוינת.
   אם הפריט קיים -הפונקציה מחזירה את האינדקס שלו, אם הפריט לא קיים – הפונקציה מחזירה -1.
   קלטו מהמשתמש פריטים ואיבר לחיפוש, העבירו לפונקציה והדפיסו את המיקום.
   לא לשכוח תיעוד.

2. כתבו קובץ בשם TrigoID.py המכיל את המחלקה TrigoID.
   במחלקה, הגדירו משתנים עבור הפונקציות הקיימות sin, cos, tan ממודול math.
   כתבו פונקציות לחישוב:
   a. סינוס של מספר נתון.
   b. קוסינוס של מספר נתון.
   c. טנגנס של מספר נתון.
   בעמוד הבא נתונות 3 טבלאות של זהויות טריגונומטריות, בחרו 3 זהויות וכתבו לכל אחת מהן פונקציה המחשבת את הזהות הטריגונומטרית.
   כתבו קובץ חדש אליו תייבאו את המודול TrigoID שכתבתם, והראו דוגמאות הרצה לפונקציות שכתבתם.

3. הוסיפו exceptions למחלקה TrigoID.

4. שימו לב: יש קשר בין שני הסעיפים הבאים:
   a. כתבו פונקציה בשם convert_str_to_int המקבלת מחרוזת (str) וממירה אותה למספר (int).

   b. כתבו פונקציה נוספת בשם calc_log המקבלת מספר x ומחשבת את הערך log(x).

   כעת, השתמשו בשתי הפונקציות ע"מ לחשב ערך של log(s) עבור מחרוזת s המתקבלת מהמשתמש.
   שימו לב: התוכנית אמורה לשרוד עבור כל קלט!

חיפוש בינארי:

חיפוש בינארי הוא אלגוריתם לחיפוש מיקומו של איבר במערך ממוין.

בהינתן מערך ממוין, האלגוריתם בודק את האיבר האמצעי שבמערך. אם הוא האיבר המבוקש –האלגוריתם מחזיר את מיקומו, אחרת, אם האיבר המבוקש קטן יותר, האלגוריתם ניגש לחלקו השמאלי של המערך ובודק את האיבר האמצעי וחוזר חלילה עד למציאת האיבר המבוקש.

זמן הריצה יורד מ-O(n) ל-O(log(n)).

בפייתון ניתן להשתמש במודול bisect_left של הספריה bisect ע"מ להריץ את האלגוריתם.

דוגמא למציאת מיקומו של איבר 2:

1	2	3	4	5	6	7	8	9	10

1	2	3	4	5

1	2	3

זהויות טריגונומטריות:

נלקחו מתוך אתר ויקיפדיה.

שיקוף דרך $\theta = 0$	שיקוף דרך $\theta = \pi/2$	שיקוף דרך $\theta = \pi$ [הסתרה]
$\sin(0-\theta) = -\sin\theta$	$\sin(\frac{\pi}{2}-\theta) = +\cos\theta$	$\sin(\pi-\theta) = +\sin\theta$
$\cos(0-\theta) = +\cos\theta$	$\cos(\frac{\pi}{2}-\theta) = +\sin\theta$	$\cos(\pi-\theta) = -\cos\theta$
$\tan(0-\theta) = -\tan\theta$	$\tan(\frac{\pi}{2}-\theta) = +\cot\theta$	$\tan(\pi-\theta) = -\tan\theta$
$\csc(0-\theta) = -\csc\theta$	$\csc(\frac{\pi}{2}-\theta) = +\sec\theta$	$\csc(\pi-\theta) = +\csc\theta$
$\sec(0-\theta) = +\sec\theta$	$\sec(\frac{\pi}{2}-\theta) = +\csc\theta$	$\sec(\pi-\theta) = -\sec\theta$
$\cot(0-\theta) = -\cot\theta$	$\cot(\frac{\pi}{2}-\theta) = +\tan\theta$	$\cot(\pi-\theta) = -\cot\theta$

הזהות הטריגונומטרית הפיתגוראית	$\sin^2\theta + \cos^2\theta = 1$
זהות היחס	$\tan\theta = \frac{\sin\theta}{\cos\theta}$

הזזה ב־$\frac{\pi}{2}$	הזזה ב־π (המחזור של tan ו־cot)	הזזה ב־2π (המחזור של sin, cos, csc ו־sec) [הסתרה]
$\sin(\theta+\frac{\pi}{2}) = +\cos\theta$	$\sin(\theta+\pi) = -\sin\theta$	$\sin(\theta+2\pi) = +\sin\theta$
$\cos(\theta+\frac{\pi}{2}) = -\sin\theta$	$\cos(\theta+\pi) = -\cos\theta$	$\cos(\theta+2\pi) = +\cos\theta$
$\tan(\theta+\frac{\pi}{2}) = -\cot\theta$	$\tan(\theta+\pi) = +\tan\theta$	$\tan(\theta+2\pi) = +\tan\theta$
$\csc(\theta+\frac{\pi}{2}) = +\sec\theta$	$\csc(\theta+\pi) = -\csc\theta$	$\csc(\theta+2\pi) = +\csc\theta$
$\sec(\theta+\frac{\pi}{2}) = -\csc\theta$	$\sec(\theta+\pi) = -\sec\theta$	$\sec(\theta+2\pi) = +\sec\theta$
$\cot(\theta+\frac{\pi}{2}) = -\tan\theta$	$\cot(\theta+\pi) = +\cot\theta$	$\cot(\theta+2\pi) = +\cot\theta$