

שפת תכנית

:BNF ג' כף -**BNF**

The symbol `::=` means *is defined as*

The symbol `|` means *or*; it separates alternatives, e.g.

`<addop> ::= + | -`

ארצנו פהווטין

- `<integer> ::= <digit> | <integer> <digit>`
or
`<integer> ::= <digit> | <digit> <integer>`
- [] enclose an optional part of the rule
 - Example:
`<if statement> ::= if (<condition>) <statement> [else <statement>]`
- { } mean the enclosed can be repeated any number of times (including zero)
 - Example:
`<parameter list> ::= ()`

$$\quad \mid (\{ <\text{parameter}>, \} <\text{parameter}>)$$

3.1.2 נתחן (2019 אוקטובי 5.0.5.0)

(2) (14 נק') כתוב דקדוק BNF לביטוי שהוא שורה חוקית בתוך טבלת מספרים.

לדוגמה,

5 | 27 | 188 |

line ::= | number | number | number |

נום

not or

number ::= _ _ ddzdz | _ _ _ ddzdz | _ _ _ _

d ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

ddzdz ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

יש לשים לב שבעל שורה יופיעו 3 מספרים בדיק. המספרים יכולים להיות חד / דו / תלת ספרתיים. המספרים מוצמדים למשנן ותחומים בתחום הטור שרווחבו קבוע (5 תווים, יכול רוחחים). שימוש לב שהקו האנכי הוא חלק מהביטוי, זהו הקו המפריד בין העמודות בטבלה וה מרחק בין הקווים האנכיים שווה. (יש להגידתו רוחכתו '_'). המספרים לא יכולים להיות עםAPS מוביל.

פתרון:

```

<line> ::= <column> <column> <column> <vertical>
<column> ::= <vertical> <space> <3digitnum> | <vertical> <space> <space>
          <2digitnum> | <vertical> <space> <space> <space> <digit>
<3digitnum> ::= <2digitnum> <digit> | <2digitnum> <zero>
<2digitnum> ::= <digit> <digit> | <digit> <zero>
<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<zero> ::= 0
<space> ::= _
<vertical> ::= |

```

-Binding נון אולון גאנן זאן
 - Static Allocation (גאנטאג שאנטער גו נלאן זאנט)
 - Dynamic Allocation (גאנטאג שאנטער נלאן זאנט)

(* נפירון נסם (אך רון וט ו סקופ) (מידורם - ספקות)

גאנטאג פונט פיריהה פלאן אינון גאנט
 name | סוכן | כרעהת

פונט נטען אהונגה, אהונגה
 אוניק אהונגה פונט גו אינון
 name | סוכן | כרעהת

local — LEOB
 ↘ ↘ ↘
 extended / global
 built in

אינון זע השפה פאנט
 name | סוכן | כרעהת

```
>> i=8
>> foo():
>
>>
```

נפירון (גיא) לא סמ' גאנטאג גאנטאג
 (גיא) זען אוקט אנטיקון זאנט פונט אהונגה
 לינטן קאנט, ואילך garbage collection זאנט נלאן
 זאנטן כראטן זען אוניק קאנט גאנטאג.
 נראטאג.

פאנטאג שאנטער נסם:

Upper() = הפיכת אותן קטנה לגודלה

Type() = מחזירה את סוג המשתנה

Isinstance(5,int) = Boolean בודקת האם המשתנה מהסוג זהה , מחזירה

join([text_list]) = הפיכת רשימה למחרוזת

Rd.randint(0,80) = מספרים רנדומליים בין טווח מסויים

Text.split('delimiter') = פיצול טקסט , ברירת מחדל לפצל ברווח

Round() = פונקציה מעגלת מספרים עשרוניים

Str*3 = הכפלת מחרוזת - שרשור 3 פעמים

Str[2:5] = רק יחת תת מחרוזת במקומות 2 עד 5 , אם נשמייט את 2

از זה יהיה מההתחילה עד 5 , אם נשמייט את 5 זה יהיה מ2 עד הסוף

Ord(char) = מחזירה מספר המציג את הערך האסקי של הצער בפנים

In range(num_start, num_finish, **קפיצות**)

וְאַתָּה נִשְׁתַּחֲווּ

לפוך כל דיאלוג נקבע בפונקציית `list`, ש带回ה מערך של כל המילים שבסעיף.

אפשר לתרגם פונקציית `count` מ-`C#` ל-`Python` באמצעות הפקה של `list.index`:

```
def count(list, x):
    count = 0
    for i in range(len(list)):
        if list[i] == x:
            count += 1
    return count
```

לפיכך list. reverse() פועל כמו list. Sort()
ולפיכך list. Sort(reverse=True)

כלי של גיבובים list^* X נקראים כפלי נטולות X

היכן לשים פסיבר : Str list מעתה כתוב Str list : ['h','e','l','l','o'] \Leftarrow list += 'hello' נPATCHING

למגר רקט נמלט x :
המוצר רקט נמלט x :
list.remove(x)
המוצר רקט נמלט x :
for i in range(list.count(x)): list.remove(x)
list.remove(x)

(הצטט אסמן נציגים:) list.pop()

• **list - x** מוחזר אל הערך המקורי ב-**del list[x]**

לפניהם ריתם פצע (באותן מקרים), וכך הם נקבעו. סעיפים 2 ו-3 של פערון |
list1 = list2 - ערך אבסולוטי יכול למכהן ככלי, בעוד ש-
list1 = list2.copy() - הגדיר מחדש גורם

לפנינו מופיעים מילים דומות: tuple ו-tuple.tuple.tuple

tuple (tuple) מוגדר כפונקציה שמייהןtuple.tuple()

tuple (tuple) מוגדר כפונקציה שמייהןtuple.tuple()

tuple = ()
tuple = tuple()

tuple (tuple) מוגדר כפונקציה שמייהןtuple.tuple()

tuple (tuple) מוגדר כפונקציה שמייהןtuple.tuple()

-tuple(x) מוגדר כפונקציה שמייהןtuple.tuple()

-tuple(x) מוגדר כפונקציה שמייהןtuple.tuple()

-tuple(x) מוגדר כפונקציה שמייהןtuple.tuple()

(tuple(x) מוגדר כפונקציה שמייהןtuple.tuple(x))

c=3, b=2, a=1 \Leftarrow a,b,c = (1,2,3)

In [28]: 1 values = [(1, 10), (2, 20), (3, 30), (4, 40)]

In [29]: 1 for n1,n2 in values:
2 print(n1, n2)

1 10
2 20
3 30
4 40

: tuple (tuple)

tuple unpacking

Enumerate
פונקציית enumerate() מודפסת אוסף של צמדים (index, item) של אוסף.

1 numbers = [1,2,3,4,1,2,4,3,4,1]

1 for index, num in enumerate(numbers):
2 if num > 3:
3 print(num, index)

4 3
4 6
4 8

zip (tuple) מוגדר כפונקציה שמייהןtuple.zip(*args)

zip

1 days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]
2 calls = [23,13,26,41,11,31,12]
3
4 list(zip(days,calls))

[(('Sunday', 23),
('Monday', 13),
('Tuesday', 26),
('Wednesday', 41),
('Thursday', 11),
('Friday', 31),
('Saturday', 12))]

4 for day_name, calls_number in zip(days,calls):
5 print(f'We had {calls_number} calls in {day_name}')

We had 23 calls in Sunday
We had 13 calls in Monday
We had 26 calls in Tuesday
We had 41 calls in Wednesday
We had 11 calls in Thursday
We had 31 calls in Friday
We had 12 calls in Saturday

ן - Dictionaries

{Key, Value}

$d = \{ \}$ ik / $d = \text{dict}()$ כטבָּה נְבָנִית

וינט פונקציית פודט מילון d - $d[\text{key}] = \text{value}$ (הנוסף ב- d)

```
1 clients = {100: {'first name': 'Rick',
2   'last name': 'Sanchez'},
3 200: {'first name': 'Peter',
4   'last name': 'Griffin'},
5 300: {'first name': 'Homer',
6   'last name': 'Simpson'}}
```

```
1 clients[100]['last name']
```

'Sanchez'

כטבָּה נְבָנִית אֲזַעַן - $\text{new_d} = d$ - נתקון נְבָנִית

איך רוחים נאכליות:

$d[\text{new_key}] = x$ פולר מפה תעוז נאכלי

(הכרת X גורכו)

ואם גנטה הלה (ויאם קה נאכליות) או תולק של T/F - $\text{new_key} \in d$

גרופר אקס סומן:

$d.pop(\text{new_key})$

נקה גנטה הלה (ויאם קה נאכליות) ועתיקות שולץ. Key שולץ נאכליות נאכליות del d[new_key]

נקה גנטה הלה (ויאם קה נאכליות) del d

נקה גנטה הלה (ויאם קה נאכליות) del d

-תרות שולץ שולץ מפה גנטה X d.get(x)

לעומת שולץ שולץ מפה גנטה (ולא שולץ שולץ מפה גנטה)

d.keys() - קבוצה שולץ שולץ מפה גנטה
d.values() - קבוצה שולץ שולץ מפה גנטה

(key, value), tuples של מילון רוחה נאכליות (ולא שולץ שולץ מפה גנטה)

```
1 for i, food in enumerate(rest):
2     print(i, food)
```

X שולץ שולץ מפה גנטה יתרכז, d -> x שולץ שולץ מפה גנטה - d.setdefault(x, y)

0 Humus
1 Pizza
2 Salad
3 Hamburger
4 Chips
5 Orange Juice
6 Beer
7 Wine

```
1 for food, price in rest.items():
2     print(f'The price of {food} is {price} NIS.')
```

The price of Humus is 15 NIS.
The price of Pizza is 25 NIS.
The price of Salad is 12 NIS.
The price of Hamburger is 35 NIS.
The price of Chips is 8 NIS.
The price of Orange Juice is 13 NIS.
The price of Beer is 17 NIS.
The price of Wine is 30 NIS.

```
1 sorted(rest.items(), key = lambda t: t[1], reverse = True)
[('Pizza', 45),
 ('Hamburger', 35),
 ('Wine', 30),
 ('Beer', 17),
 ('Orange Juice', 13),
 ('Humus', 12),
 ('Salad', 12),
 ('Coffee', 10),
 ('Chips', 8),
 ('Tea', 5)]
```

Sorted(d, key=0)(*)

ויאם קה נאכליות נאכליות

len שולץ

ויאם קה נאכליות שולץ שולץ מפה גנטה (ולא שולץ שולץ מפה גנטה)

מייל ניגון: מפה גנטה

```
1 for key in sorted(rest):
2     print(key, rest[key])
```

Beer 17
Chips 8
Coffee 10
Hamburger 35
Humus 12
Orange Juice 13
Pizza 45
Salad 12
Tea 5
Wine 30

```
1 sorted(rest)
```

Beer
Chips
Coffee
Hamburger
Humus
Orange Juice
Pizza
Salad

ויאם קה נאכליות נאכליות

ויאם קה נאכליות נאכליות (ולא שולץ שולץ מפה גנטה)

new_d = d.copy()

ב-**Sets** (sett) מוגדרים אוסף של נתונים. אוסף סט מוגדר באמצעות פונקציית `Set()`.
לדוגמה: `S = Set([1, 2, 3])`

Set of x numbers $S.\text{add}(x)$ - Set of p s numbers

1 {1,2,3,4,3,2,1,2,3, 100 > 200, 200 > 300}
False 1 2 3 4 1) false false

אך אם נשים מטרת ה-Set כפולה, מטרת ה-*tuple* כפולה.

`s.remove(x)` - Set N
`s.discard(x)`

```
1 unique_numbers = set([1,2,3,2,1,2,3,4,3,3,2,5,7,8,9,8,7,6,5,4,3,2,1,2,3,4,5,6])  
1 unique_numbers  
set([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

S.copy() : גנטים
S.clear() : Set עקי'
ר'יעו'ן כפליית אסאהה :

```
1 dc_comics = {'Superman', 'Batman', 'Joker', 'Wonder Woman', 'Deadshot'}
2 bad_guys = {'Joker', 'Shredder', 'Rick Sanchez', 'Deadshot', 'Magneto', 'Captain Hook'}
```

: Set (אוסף נתונים)

```
1 dc_comics.union(bad_guys)
{'Batman',
'Captain Hook',
'Deadshot',
'Joker',
'Magneto',
'Rick Sanchez',
'Shredder',
'Superman',
'Wonder Woman'}
```

הנגזר מ-`s.union(new_S)` (ההויר) בהנגזר מ-`s.update(new_S)` (ההויר)
ולכן `s.intersection(new_S)` הויר מ-`s.intersection_update(new_S)` (ההויר)
ולכן `s.intersection_update(new_S)` הויר מ-`s.intersection(new_S)` (ההויר)

new_S.difference(new_S) : new_S.difference(new_S)

הנחת רכינה בפ' ק' (האורים) (הנחתות) : S. Symmetric_difference(new_S)

T/F S. isSubset(new_S) : הבודק אם סט נ' יתפרש כSubset של סט נ' נכון

T/F S. isSuperset(new_S)

הפייננסים נרחבו Set 3 (בנוסף ל-Set 2) על מנת לכסות את הלקוחות שמשתמשים ב-

היתוך נאנו $\Leftarrow T/F$

5. `isdisjoint(new_s)` : פונקציית `הירקן` בנארה בנארה בנארה

List Comprehensions

ביק שפוך גיאץ' ולייןר הנקראת דיאינט, נארה נאבר. תומתך גראקי, נארה (גיאץ' דיאינט)

```
1 prices = [10, 14, 56, 34, 87, 34, 12, 95]  
  
1 prices_in_shekels = []  
2 for price in prices:  
3     prices_in_shekels.append(price * 3.5)  
4  
5 prices_in_shekels  
[35.0, 49.0, 196.0, 119.0, 304.5, 119.0, 42.0, 332.5]  
  
1 del prices_in_shekels  
  
1 prices_in_shekels = [price * 3.5 for price in prices]  
2 prices_in_shekels  
[35.0, 49.0, 196.0, 119.0, 304.5, 119.0, 42.0, 332.5]
```

כארה ד
list comprehensions

כינופר דיאינט

או וויה בזען else
רכז גאנטיים או אם if קהונת בזען

above_120 = [n * 1.2 if n > 120 else n for n in numbers]

אילו (ENDIAN) אוניברסיטת מילוי

```
1 text = 'The past can hurt. But the way I see it, you can either run from it, or learn fro  
1 vowels = 'aeiou'  
1 ''.join(['*' if letter in vowels else letter for letter in text])  
'Th* p*st c*n h*rt. B*t th* w*y I s** t, y** c*n **th*r r*n fr*m *t, *r l**rn fr*m *t.'
```

בזען סוף כארה

כט פטת שטניאס זויגווע
האילו הומלא פיעל פוכמי

```
temp = [rd.randint(0,38) for n in range(30)]  
  
['Cold' if t < 15 else 'Nice' if t < 25 else 'Hot' for t in temp]
```

פליזה נחנת if,else

... Not ,in זהה קולטונג נאלה בזען if,else

Dictionary Comprehensions

בכך ניתן ליצור אוסף נתונים מסוים רק אם הוא מקיים נספח ב-

```
1 squares = {n: n**2 for n in range(1,11)}
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100}
```

באותה החלטה: (*)

```
1 words = {word: len(word) for word in fc.split()}
```

```
2 words
```

```
{'The': 3,  
'first': 5,  
'rule': 4,  
'of': 2,  
'Fight': 5,  
'Club': 4,  
'is': 2,  
'you': 3,  
'do': 2,  
'not': 3,  
'talk': 4,  
'about': 5,  
'Club.': 5}
```

: סדר פונקציית (*)

fc = 'The first rule of Fight Club is you do not talk about Fight Club.'



(ב) גיבר נפקחות (אלה): (*)

```
1 values = {1: 'aa', 2: 'bb', 3: 'cc', 4: 'dd', 5: 'ee', 6: 'ff'}
```

```
1 {v: k for k,v in values.items()}
```

```
{'aa': 1, 'bb': 2, 'cc': 3, 'dd': 4, 'ee': 5, 'ff': 6}
```

גיבר סור גיאוגרפיה: (*)

```
1 {v: k for k,v in values.items() if k % 2 == 0}
```

```
{'bb': 2, 'dd': 4, 'ff': 6}
```

(ז) גיבר נפקחות ועכבר:

(ז) גיבר נפקחות ועכבר: (*)

drinks = {'Cola': 5, 'Coffee': 8.5, 'Tea': 6, 'Beer': 12, 'Wine': 15} : גיבר גיבר כה נחיה גנאה נספח: (*)

גיבר ↑

```
1 drinks = {name: {'NIS': price, 'USD': price / 3.52} for name, price in drinks.items()}
```

Variable Scope

ארהכ / תמיון נאות(ו)

(*) ר'ין פ'ם כהו כתור דג, ס'מ' כתה לא דג ד'ר' ה'ל'ר'ן ס'מ' ג'ט'ג'ה ג'ט'ג'ה נה.

כג'אל או נזרחים נערעה פ'אי מתק פ'אנ', כה ה'ל'ק' ה'ל' א'ל'ה צ'ו'ן י'ג' ר'ית' ג'ל'ק'י'ו'ס ס'מ' ג'ע'יה'ס ג'ע'יה'ס כו' ה'ל'ל'ר'ה ז'ג' פ'אנ'ג'ה.

או' פ'א'ל'ע'ה נ'ז' א'ל'ע' ג'ל'ל'ג'ה.

המונח **Outer** מתייחס ל~~לפניהם~~ **Inner** Scopes (ריצויים) שמייצגים תחומי גלגולים.

הפוך רחבה level - אינטראקטיבית (ב) scope (א) מוגבלת

כדי לאירועים מסוימים ב-`global` scope נויצת, או על מנת לארחם ב-

פתקון status נשלח (וכן), וlain'aici'i כחיק הפלגה, (כג ש'י) 2'יכן סג'ו ישרה (אלוון זעט נשלח (וכן) כחיק הפלגה, (כג ש'י) סג'ו ישרה).

```
def some_func():
    global status
    if status:
        print('Running...')
        status = False
    else:
        print('Can not execute right now...')

some_func()
running...
status
```

func **c** (פונקציית **c**)

```
def func_c():
    def func_d():
        nonlocal y
        y = 50

    y = 10
    print(f"y before func_d y={y}")
    func_d()
    return f"y after func_d {y}"

print(func_c())

```

output:

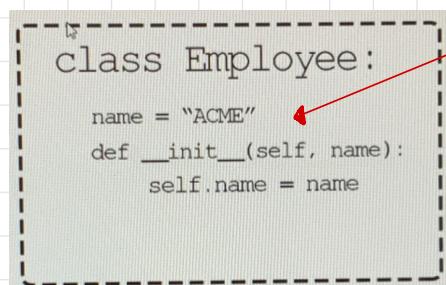
```
y before func_d y=10
y after func_d 50
>>> ===== RES =====
```

הנפקה, הרכבה

הנפקה (pull) כוננה ל-

הרכבה (push) כוננה ל-

-class Variables
- instance Variables



```

class myClass:
    def __init__(self):
        self.x = "public"
        self._x = "protected"
        self.__x = "private"
    
```

class - ה קורן יופיע ב- self.x

```

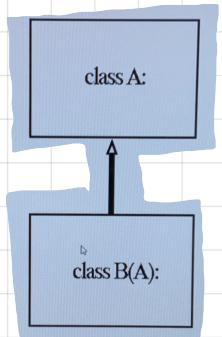
class A:
    pass

class B(A):
    pass

a = A()
b = B()
    
```

A של B מושפע מ- A-N ב- B

כדי אונטג'ון - יונט



```

class citizen:
    def __init__(self, nm):
        self.firstname = nm

    def whois(self):
        print("i am", self.firstname)

class student(citizen):
    def __init__(self, nm, stuid):
        citizen.__init__(self, nm)
        self.studentID = stuid

    def whois(self):
        print("i am", self.firstname, "i am a Student #", self.studentID)

s = student("bob", 1234)
c = citizen("pol")
    
```

הנפקה פול' כוננה ל- selffirstname (ולא self)

```

class PAPA:
    def eatHotSpice(self):
        print("mmmm i love hot spice!")

class MAMA:
    def dontEatDirtyFood(self):
        print("is this dirty? i dont eat dirty foodz...")

class SON(PAPA, MAMA):
    pass

son = SON()

son.eatHotSpice()
son.dontEatDirtyFood()
    
```

• iterator

collection (list, dict, str, tuple) - iterable

וילע דהראם אם / היבא לתשואת כליאן האילן Collection - next() collections - next() .

Stop Iteration ↵

הנתקות iter() מ-list

: iterable | כורס אוסף

```
Class A():
    def __iter__():
        def __next__():
            ...
            raise stopIteration
```

ללא ריכם פלוי א' 88(1) A(A) = A(a)

Creating Our Own Iterator

- Here is how we can add iteration to our own classes:
 - We define an `__iter__()` method which returns an object with a `__next__()` method
 - We can implement `__next__()` inside the class, and simply return `self` as the iterator

```
class Reverse:
    def __init__(self, data):
        self.data = data
        self.index = len(data)
    def __iter__(self):
        return self
    def __next__(self):
        if self.index == 0:
            raise StopIteration
        self.index = self.index - 1
        return self.data[self.index]
```

```
>>> rev = Reverse('spam')
>>> iter(rev)
<__main__.Reverse object at 0x00A1DB50>
>>> for char in rev:
...     print(char)
...
m
a
p
```


Generator גנרטור

הנוצר נוצר נוצר - `__init__`

הנוצר יתבצע נוצר נוצר - `__iter__`

`StopIteration` יתבצע אם הולך לא יכול יותר לשוב חזרה ל `next` ב- `__iter__` - `__next__`

```
class MyRange:

    def __init__(self, start, end):
        self.value = start
        self.end = end

    def __iter__(self):
        return self

    def __next__(self):
        if self.value >= self.end:
            raise StopIteration → ! סוף!
        current = self.value
        self.value += 1
        return current
```

הנוצר יתבצע כרך סוף

```
def my_range(start, end):
    current = start
    while current < end:
        yield current
        current += 1
```

```
18
19 nums = MyRange(1, 10)
20
21 for num in nums:
22     print(num) → 1-9 : N סוף 003,
```

: סוף

Decorator

כונן "_decorator" - הוסף פונקציה שאנדרטורה פועל' אחורית אחתייה פועל' (פונקציית).

כפועתיה של הפונקציה נזקפתה על הפונקציה שפועתיה.

אפשרה נחוצה את הפונקציה אל הפונקציית, נאלה הפעלה רצף גיבובים ותתבצע.

בכל גל', גל' וначיה.

בכל גל' גל' וначיה.

בכל גל' גל' וначיה.

```
def dec(func):
    def wrap():
        print("before")
        func()
        print("after")
    return wrap()

def fool():
    print("hello")
```

הו בז' פועלן ←
hello

dec(fool) → before
hello after

: 0.03

הו בז' פועלן ←
func(fool) נאלה בז' בז' נאלה בז'

הו בז' לפ' זה ריתן פונקציון
⇒ print(dec(fool)())
func(fool) נאלה בז' הפוך' לתוכלו

"C:\Program Files\Python36\python.exe" C:/Users/afe
<function dec.<locals>.wrap at 0x00000000268C0D0>
before
hello
after

↓

ה' גל' ריתון בז' נאלה בז'

fool נאלה בז'

ה' קומפליקט sk)
fool()

≡ dec(fool)():
הו בז' :

```
@dec
def fool():
    print("hello")
```

→ now fool

ריתן פועלן כוכ נאלה
בקומפליקט

@dec1
@dec2
↓

ה' ריתן ווען נאלה נאלה
dec2 נאלה (נק' dec1)

@dec1
@dec2
↓

Decorator of Decorators

האיים נאמרים בסעיפים של tuple Positionals (ירוק פג'מ'ר 2) Key words (ירוק פג'מ'ר 2) נספ'ל חנוך (אך מ'ן גונן מ'ן כוכ'ת כה) - *args - **kwargs

```
def dec(func):
    def wrapper(*ar, **kw):
        val = func(*ar, **kw)
        if val %2==0:
            return val
        else:
            return val+1
    return wrapper

@dec
def fool(a,b):
    return a+b

print(fool(3,4))
print(fool(3,3))
print(fool(2,4))
```

8
6
6

-cnr3

כל הדרישות שבקה בפ' חנוכ'ה
הפ'ן, הפעיל.

איך מ'ן אחראי ←
על הערך

cnr3 סעיף 3) סעיף 3) סעיף 3)

```
def dec(func):
    i=0
    def wrapper(*ar, **kw):
        nonlocal i
        i+=1
        print("i = ",i)
        val = func(*ar, **kw)
        if val %2==0:
            return val
        else:
            return val+1
    return wrapper

@dec
def fool(a,b):
    return a+b

@dec
def foo2(a,b):
    return a*b
```

כדי רצף תלויה כה פ' נאסר
הפ'ן 1-ב' נספ'ל $i \rightarrow$ 1 כה גורילה
הפ'ן foo2-ב' רצף פ' נספ'ל
ויהי מ'ן אהנו חfine ⇒
רכ'ן צו'ם שול'

Decorator Модель

- `__call__(self)` - מילוי ה-`func` ב-`__call__(self)`

```
class class_wrapper:  
    def __init__(self, func_to_wrap):  
        self.func = func_to_wrap  
  
    def __call__(self, *args, **kwargs):  
        print("extending function with class")  
        self.func(*args, **kwargs)  
  
@class_wrapper  
def a_func3():  
    print("I'm a first class function!")  
  
@class_wrapper  
def func_with_arg3(msg):  
    print("func_with_arg prints:", msg)
```

: a_func3 nlc f(x0))

```
>>> a_func3()  
extending function with class  
I'm a first class function!
```

Lambda λ

המכירך (ג'ונאטה אנטהאער) אה בתייר פועל' אוועזיאער - יטיר שער' אונז' מלהר גה שא נטוויה ליכלה ווועז
זאייעס (כשאער מאה' אקייס (טראה מאטצעה).
ווען - חוטם כתיכת, (וועק, זויז קאיה) ווועז.
רעלעה לאט נטעלע כוואל ריטה ג'אנכיז פוינ', (וואלונר) גיטוונ', אומחר, או שרכיה גודז איקטעה שער'
בונגע לאט פערע' צאנכט.

את קרען אתלויס : לא זעפער אונדזיאן

אנרג:

```
1 money = {'Tokyo': 400,
2   'Nairobi': 590,
3   'El Profesor': 650,
4   'Denver': 530,
5   'Rio': 270,
6   'Bogotá': 340,
7   'Berlin': 480}

1 sorted(money.items(), key = lambda t: t[1], reverse=True)
[('El Profesor', 650),
 ('Nairobi', 590),
 ('Denver', 530),
 ('Berlin', 480),
 ('Tokyo', 400),
 ('Bogotá', 340),
 ('Rio', 270)]
```

(א) מון אובייס :

Map

פועל', פאילכת גויז פאנפער אונכיס אונז'ים פאנז'ים געלאט. אונלאג פאנז'ה קאנז'ה גויז גאנז'ה אונז'ים געלאט. וויה ליגז גאנז'ה אונז'ים געלאט.

פונקציית map מקבלת פונקציה ורשימה. נוצרת רשימה חדשה, שהיא התוצאה של הריצת הפונקציה על כל אחד מאיברי הרשימה המקורי.

במילים אחרות, לכתוב `new_list = map(func, old_list)` זה כמו לכתוב:

```
new_list = []
for element in old_list:
    new_list.append(func(element))
```

כפי שבתוח ניחשתם, את `func` אין צורך להגדיר ממש, אלא ניתן להכניס lambda כרצוננו.

לדוגמא, ניקח רשימת מספרים ונרצה לכפול כל מספר פי 2 ולהוסיף 1:

```
old_list = [3, 6, 1, 7, 5]
print(set(map(lambda x: 2*x+1, old_list)))
```

ונקבל {3, 7, 11, 13, 15}. שימושו לב שלצורך ההדפסה הפכנו את תוכחת המיפוי לאובייקט מטיפוס `set`, שנitin להדפסה.

אנחנו לא יכולים להסתפק ברשימה אחת. אפשר לכתוב פונקציית lambda שמקבלת יותר פרמטר אחד ולהעביר ל-map כמה מוגדרות של רשימות. לדוגמה, ישן שתי רשימות ואנחנו מעוניינים לכפול אותן זו בזו (מה שנקרא "מכפלה וקטורית" במתמטיקה):

```
list1 = [1, 2, -5, 6]
list2 = [2, -1, 3, 4]
```

ונקבל {2, -2, 15, -24}.
{2, -2, 15, -24}

reduce

אוצרת פע' אוסף אוסף וგאנז / פג'הית אולו גאנז נאכ'.
from functools import reduce
(בכל ג'יכא אט' הפונק' צי)



- (*) גפוען שערן צי
- (1) ח'יכת גאנט צו זוכיך
- (2) ח'יכת גאנטלי ציק זוכיך

הfonקצייה reduce מקבלת פונקציה ורשימה ומחזירה ערך יחיד. הערך זהה הוא התוצאה של ביצוע הפונקציה על איברי הרשימה שוב ושוב עד שנותר ערך יחיד. אפשר לחשב על בסיס reduce בטור map שימוש שמשיר להבצע כל עוד אוורך הרשימה החדשה שנוצרת גדול מ-1.

לדוגמה, יש לנו רשימה ואנחנו רוצים לחשב את הסכום של האיברים שלה:

```
old_list = [2, 3, 4, 7, 8, 10]  
print(reduce(lambda x, y: x+y, old_list))
```

אפשר לדמיין שבכל שלב נוצרת רשימה חדשה, שבה האיבר הראשון הוא סכום שני האיברים בראשימה המקורי:

- [5, 4, 7, 8, 10] •
- [9, 7, 8, 10] •
- [16, 8, 10] •
- [24, 10] •
- [34] •

כאשר הגיעו לרשימה באורך 1 החישוב ייעצר ויוחזר הערך שנותר בה.

Filter



- (*) גפוען שערן צי
- (1) ח'יכת גאנט צו זוכיך
- (2) ח'יכת גאנטלי ציק זוכיך

אוצרת פע' אוסף צפוי איברטון קידטווון צואוונז נאכ'ויאם.
אכ'רליג צפוי איבט נאכ'רליג גאנט גאנט נפוען, עטטע גה, וו' זט' גפוען,
גתקלה ענד צפוי איזו צק filter תראוו צו זוטר נאכ'ר גאנט לא'ינט.
נכ'ק (פ'לט) אנטאלסיא (ג'ג') (ט'א'ס).

פונקציה זו נועדה לסתן (to) רק איברים רלבנטיים מתוך רשימה קיימת. לשם כך, הפונקציה filter מקבלת פונקציה ורשימה, בודקת על כל אחד מאיברי הרשימה האם הפונקציה מחזירה עליו True ומחזירה רשימה רק של האיברים שהחזירו True. כך, לדוגמה –

new_list = filter(func, old_list)

```
new_list = []  
for element in old_list:  
    if func(element) is True:  
        new_list.append(func(element))
```

לדוגמה, נרצה לקחת רשימה ולייצור ממנה רשימה חדשה רק של המספרים הזוגיים:

```
old_list = [2, 3, 4, 7, 8, 10]  
print(set(filter(lambda x: x % 2 == 0, old_list)))
```

הסבר: קודם כל אנחנו יוצרים פונקציית lambda, שמחזירה את הערך הבויאני של הביטוי $x \% 2 == 0$. לאחר מכן

אנחנו מכניסים לטור filter את הפונקציה הנ"ל יחד עם רשימה.